Similarity Models in Distributional Semantics using Task Specific Information

A thesis submitted for the degree of
Doctor of Natural Science (Dr. rer. nat.)

In the subject of Computer Science

by

Rosa Tsegaye Aga

Department of Computer Science
Information Systems and Machine Learning Lab (ISMLL)

UNIVERSITY OF HILDESHEIM, GERMANY

Winter 2018

I dedicate this dissertation to my late father, my mom, my brother and my sister, whose affection, love, encouragement and prays of day and night make me able to achieve this success.

Acknowledgements

First of all, I would like to express my sincere gratitude to my supervisors Prof. Dr. Christian Wartena and Prof. Dr. Dr. Lars Schmidt-Thieme. For all their valuable advice, patience, constant encouragement, the immense knowledge shared and appreciation throughout my work, I am eternally grateful. Their advice on both of my researches as well as on my career have been invaluable.

My PhD has been an excellent experience. Prof. Dr. Dr. Lars Schmidt-Thieme, I cannot thank him enough for everything, for his guidance, great support and kind advice throughout my PhD studies time. It was a real privilege and an honor for me to have the opportunity to work with him. Prof. Dr. Christian Wartena, as well, I cannot thank him enough not only for his academic support, but for being by my side for all the help I need from the beginning until the end to have a stable life in Germany. I would also like to express my tremendous gratitude to Dr. Lucas Drumond for being available, providing useful feedback on my work.

I am grateful to all the members of the Information Systems and Machine Learning Lab (ISMLL) at the University of Hildesheim. Thank you very much for the extensive guidance, pieces of advice and comments that they have been giving me throughout my PhD research studies. I would also like to thank my colleagues in Hochschule Hannover who have helped me to make my time so enjoyable and colorful. Thank you ALL for all the good and truly memorable times we shared together. I would like to thank especially my office mate Jean Charbonnier for checking the German translation of this dissertation Abstract.

I am especially thankful to Mr. Taiwo Dayo Ajakaye. He has enriched my PhD time in a way that words cannot express his advice and support to stay strong and positive, helped me to reach where I am right now. In addition to that, I cannot thank him enough for providing me with useful feedback on my work and for the time spent reviewing drafts of my dissertation. My sister Feven Tsegaye Aga as well, her review and valuable feedback on my dissertation were tremendous. Feven Tsegaye Aga and Mr. Taiwo Dayo Ajakaye comments have contributed a great deal to make the final document what it is today.

I am deeply thankful to my family for their love and support. A special thanks and heartfelt gratitude to my mom, Genet Seyoum. Her inspiration, guidance, sacrifices and unconditional love have been my greatest strength. Last but not least, I would like to thank my youngest brother Henok Tsegaye and youngest sister Feven Tsegaye. They were always supporting me and encouraging me with their best wishes. I would like to thank ALL my good people, I would probably not be where I am today without their inputs, guidance, encouragement, and friendship.

UNIVERSITY OF HILDESHEIM, GERMANY

Abstract

Faculty of Mathematics, Natural Sciences, Economics and Computer Science
Department of Computer Science

Doctor of Natural Science (Dr. rer. nat.)

Similarity Models in Distributional Semantics using Task Specific Information

by Rosa Tsegaye Aga

In distributional semantics, the unsupervised learning approach has been widely used for a large number of tasks. On the other hand, supervised learning has less coverage.

In this dissertation, we investigate the supervised learning approach for semantic relatedness tasks in distributional semantics. The investigation considers mainly semantic similarity and semantic classification tasks. Existing and newly-constructed datasets are used as an input for the experiments. The new datasets are constructed from thesauruses like Eurovoc. The Eurovoc thesaurus is a multilingual thesaurus maintained by the Publications Office of the European Union. The meaning of the words in the dataset is represented by using a distributional semantic approach.

The distributional semantic approach collects co-occurrence information from large texts and represents the words in high-dimensional vectors. The English words are represented by using UkWaK corpus while German words are represented by using DeWaC corpus. After representing each word by the high dimensional vector, different supervised machine learning methods are used on the selected tasks. The outputs from the supervised machine learning methods are evaluated by comparing the tasks performance and accuracy with the state of the art unsupervised machine learning methods' results. In addition, multi-relational matrix factorization is introduced as one supervised learning method in distributional semantics. This dissertation shows the multi-relational matrix factorization method as a good alternative method to integrate different sources of information of words in distributional semantics.

In the dissertation, some new applications are also introduced. One of the applications is an application which analyzes a German company's website text, and provides information about the company with a concept cloud visualization. The other applications are automatic recognition/disambiguation of the library of congress subject headings and automatic identification of synonym relations in the Dutch Parliament thesaurus applications.

Keywords: Distributional semantics, Supervised machine learning, Unsupervised machine learning, natural language processing, multi-relational matrix factorization

UNIVERSITY OF HILDESHEIM, GERMANY

Abstract

Faculty of Mathematics, Natural Sciences, Economics and Computer Science
Department of Computer Science

Doctor of Natural Science (Dr. rer. nat.)

Similarity Models in Distributional Semantics using Task Specific Information

by Rosa Tsegaye Aga

In der distrbutionelle Semantik werden abstrakte Merkmale für die Darstellung von Wörtern in einem hochdimensionalem Raum im Allgemeinen mit unüberwachten Verfahren gelernt. Die Idee ist hierbei oft, dass es eine allgemeine Darstellung für jedes Wort geben soll, die für verschiedene Zwecke im Bereich der Semantik verwendet werden kann. Die Merkmalsvektoren werden vorallem zur Berechnung der semantischen Ähnlichkeit zwischen Wörtern benutzt, wofür im allgemeinen eine einfache Kosinusähnlichkeit benutzt wird. Überwachte Lernverfahren werden hierfür nur selten benutzt.

In der vorliegenden Arbeit untersuchen wir die Möglichkeiten von überwachten Lernverfahren für verschiedene Aufgaben in der distributionellen Semantik. Die Untersuchungen beziehen sich hauptsächlich auf Aufgaben im Bereich der semantischen Ähnlichkeit und semantischen Klassifikation. Für die Experimente werden sowohl bekannte, standardisierte, wie auch neu aufgebaute Datensätze verwendet. Die neue Datensätze wurden aus Thesauri, wie dem Eurovoc Thesaurus und dem STW, abgeleitet. Eurovoc ist ein mehrsprachiger Thesaurus der vom Amt für Veröffentlichungen der Europäischen Union gepflegt wird. STW ist ein Thesaurus mit Schlagwörtern zum indexieren von Publikation aus den Wirtschaftswissenschaften, der vom Leibniz-Informationszentrum Wirtschaft gepflegt wird.

In der distributionellen Semantik werden in sehr großen Textmengen Informationen über Kontexte, in denen ein Wort vorkommt aggregiert und werden die Wörter letztendlich durch hochdimensionalen Merkmalsvektoren, die hieraus abgeleitet werden, dargestellt. Für die englischen Wörter nutzen wir den UKWaC Corpus als Datenbasis, für das Deutsch den DeWaC Corpus. Für einige Experimente werden zusätzlich Informationen aus WordNet benutzt.

Als Kontextinformation werden ausschließlich lokale Kookkurrenzwerte zu einer ausgewählten Wortmenge benutzt. Aus den rohen Kookkurrenzwerten werden in überwachten Lernverfahren abstrakte Merkmalsvektoren, die für eine bestimmte Aufgabe optimiert sind, abgeleitet. Die technische Umsetzung dieser Idee erfolgt mit Multirelationaler Matrixfaktorisierung (MRMF). Die Ergebnisse für verschiedene Aufgaben werden mit state-of-the-art Verfahren verglichen und es wird gezeigt, dass MRMF ein geeignetes Verfahren zur Integration von lexikalisch-semantischen Informationen aus verschiedenen Quellen ist.

In der vorliegenden Arbeit werden auch einige neue Anwendungen eingeführt. Eine ist eine Anwendung, die Unternehmenswebseiten analysiert und Informationen über eine Firma als Konzeptwolke darstellt. Eine weitere Anwendung ist die automatische Anreicherung eines Thesaurus mit Synonymen, die hier exemplarisch mit dem Thesaurus des niederländischen Parlaments durchgeführt wird.

Contents

List of Figures

List of Tables

List of Abbreviations

ALS	**A**lternating **L**east **S**quares
BNC	**B**ritish **N**ational **C**orpus
CD	**C**oordinate **D**ecent
CBOW	**C**ontinuous **B**ag **O**f **W**ord
DF	**D**istributional **F**eatures
LCSH	**L**ibrary **C**ongress **S**ubject **H**eadings
LF	**L**ogarithm **F**requency
LR	**L**ogistic **R**egression
KL	**K**ullback **L**eibler
PMI	**P**oint-wise **M**utual **I**nformation
PPMI	**P**ositive **P**oint-wise **M**utual **I**nformation
MRMF	**M**ulti **R**elational **M**atrix **F**actorization
SGD	**S**tocastic **G**radiant **D**ecent
STW	**S**tandard **T**hesaurus **W**irtschaft
SVD	**S**ingular **V**alue **D**ecomposition
SVM	**S**upport **V**ector **M**achine
WN	**W**ord**N**et

List of Symbols and Notations

μ	mu
α	alpha
λ	lambda
σ	Standard Deviation
$\mathbf{I}$	Identity Matrix
$\mathcal{N}$	Normal Distribution
$\mathcal{U}$	Normal Distribution
$\mathbb{R}$	The set of real numbers
X, Z, Y	matrices
X^T, Z^T, Y^T	transposed matrices
$\|X\|$	matrix norm
$\overrightarrow{V}$	Vector
v_i	Vector indexed for some purpose
$\mathbb{R}^m$	The set of m-dimensional vectors over $\mathbb{R}$
$\mathbb{R}^{nxm}$	The matrix of size n x m over $\mathbb{R}$
$A \odot B$	Hadamard (elementwise) product between vectors A and B

Chapter 1

Introduction

1.1 Semantics

Semantics is the branch of linguistics which studies the meaning of words, phrases, sentences and larger units of discourse. Meaning can be conceptual meaning and associative meaning. Semantics studies the conceptual meaning, which is the objective and conventional meaning of words. The study of meaning within linguistics can be done on a number of different levels, which are lexical semantics, phrasal semantics, sentence semantics, and discourse semantics. Lexical semantics is one of the levels that the dissertation focuses on mainly. Lexical semantics is concerned with the identification and representation of the semantics of lexical items or individual words. In addition, phrasal semantics has also been studied in the dissertation to investigate one of our lexical semantics methods' performance.

In the late 18th and early 19th centuries, Ogden and Richards studied the relationship between thoughts and things. For the relationship, they set up a triangle model which is called *Triangle of Meaning*. They published the triangle model in their book which is called *The Meaning of Meaning: A Study of the Influence of Language upon Thought and of the Science of Symbolism*. In their study, they identified that *understanding comes from within the people rather than from the words they just interpret*. As Ogden and Richards (1989) stated, a proper understanding of a word or a sign requires some sort of references being linked to it. The process of getting acquainted with a word or sign via references is called *meaning of meaning* (Ogden and Richards, 1989). The Triangle in their study shows the relationship between *words* and *thoughts*, and *thoughts* and *things*. Figure 1.1 illustrates the triangle model with the car parking example. In the figure, *symbol* represents words, *reference* represents thoughts and *referent* represents things. The dotted line in the figure between symbol and referent shows that words and things do not have a direct relation. Based on this perspective, the word is not a thing, but the word stands for the referent. The relationship between words and thoughts is more likely talking or listening, while the relationship between thoughts and things is perceiving or thinking. In Figure 1.1, the car parking example shows that the icon of no parking is a sign/symbol. The subject thinks or references the symbol as *Don't park here*, when it sees the symbol. The reference or the thinking of the subject refers to *No parking*. The sign/symbol of "no parking" does not have a direct relationship with the *No Parking* referent. The referent *No Parking* simply shows what the sign/symbol "no parking" stands for. We can see one more example to clear the idea of the triangle. For example, we have the word *Eiffel Tower*. Based on the triangle definition, *Eiffel Tower*, the building itself is a *symbol*. The subject references the symbol as the *mental idea of the Eiffel Tower*. The reference or the thinking of the subject refers to the referent *Eiffel Tower*. The referent *Eiffel Tower* shows what the sign/symbol stands for.

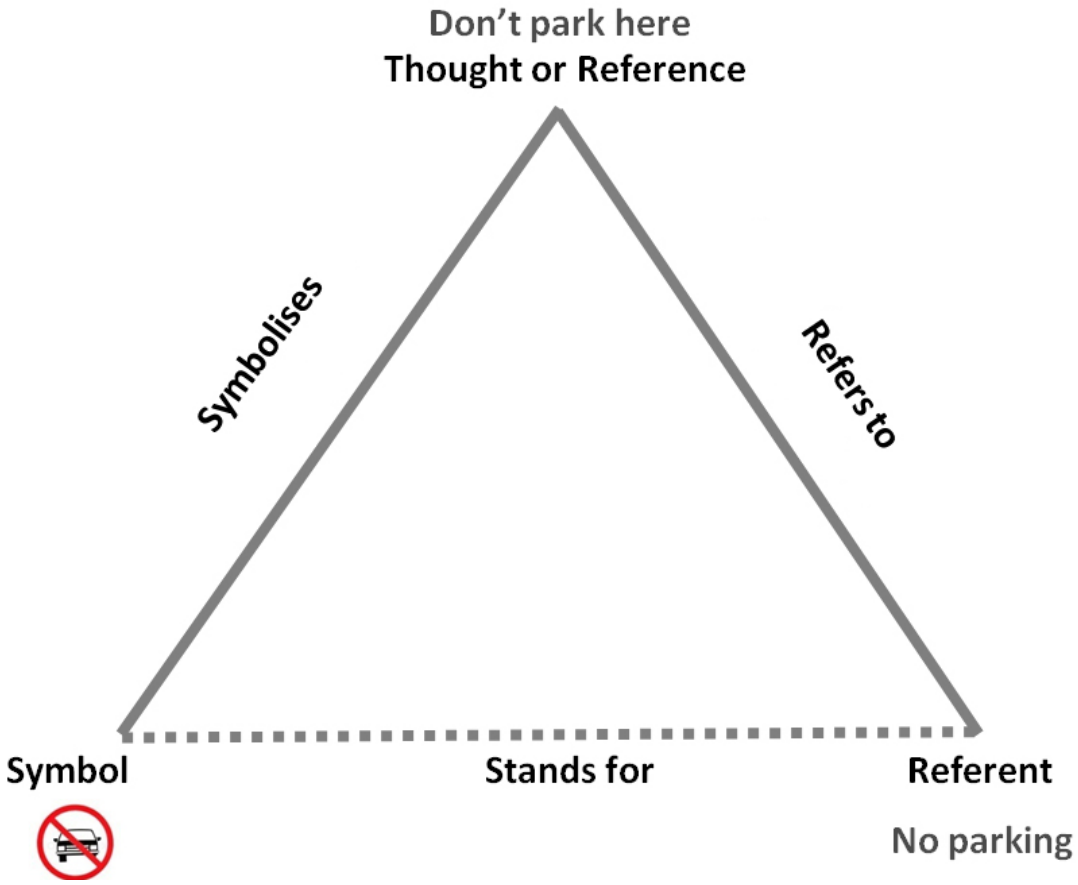

FIGURE 1.1: The Meaning of Meaning of Model by *Charles Kay Ogden and Ivor Armstrong Richards*

The study of semantics is mainly related to the subject of representation of words by reference and denotation. However, it is hard to describe the meaning of a word by the triangle model. Instead, we can use other words to describe the meaning of a word by defining their relations. This may not give the direct meaning of the word; yet it is possible to get some grip of the meaning by building semantic relations. Therefore, semantics studies the relationships between different linguistic units. The main concern in lexical semantics is how to model the meaning of a word. Some of the different linguistic units relations are as follow:

- Homonymy - words with the same spelling and/or pronunciation but different meanings. Example: root and route;
- Synonymy - words with different forms but similar meanings. Example: intelligent and smart;
- Antonymy - words with opposite meanings. Example: hot and cold;
- Hypernymy - a more broad term. Its the semantic association of being part of a higher class. Example: Animal is a hypernym of Tiger, lion and elephant;
- Hyponymy - It is the relation of inclusion. It describes a more specific term. Example: Tiger, lion, and elephant are hyponyms of the word animal;

- Co-hyponym - When separate hyponyms share the same hypernym but are not hyponyms of one another. Example: Tiger, lion and elephant are all co-hyponyms of one another and hyponyms of an animal.
- Meronymy - a word that denotes a constituent part or a member of something. Example: finger is meronym of hand;
- Metonymy - a word that is used to stand in for another word. Example: Cup, a substitute for a mug, Dish, a substitute for an entire plate of food;
- Holonymy - the relationship between a word denoting the whole and a word denoting a part of, or a member of, the whole. Example: forest is a holonym of tree (forests contain trees);
- Paronyms - words that are pronounced or written in a similar way but which have different meanings. Example: collision and collusion.

Traditionally, semantics includes the study of sense and denotative reference, truth conditions, argument structure, thematic roles, discourse analysis and the linkage of all of these to syntax. In the late 1960s, Richard Montague proposed an approach to formulating a syntax and a semantics for natural languages using formal logic, especially via higher-order predicate logic and lambda calculus. The basic aim of the approach is *to characterize the notion of a true sentence (under a given interpretation) and of entailment* (Montague, 1970). For example: *John loves Mary. She hates him*. The simple logical form of this example is: love(j,m) $\wedge$ hate(m,j). The example shows what formal logic looks like. For more examples and explanations of Montague semantics, the following resources and more can be referred to (Montague, 1970; Muskens, 1996).

In around 1982, frame semantics was proposed by Charles J. Fillmore. Frame semantics defines the meaning of a word as a coherent structure of related concepts (Fillmore, 1982). The main idea of frame semantics is finding the meaning of a word by having all essential knowledge related to the word. This means, to understand the concept of a word or to find the meaning of a word, it is necessary to understand all the related words or concepts. One of the well-known (Fillmore, 1982) frame semantic examples is the commercial transaction frame. The commercial transaction frame involves frame elements such as a seller, a buyer, goods, and money. A specific element reveals from which particular perspective the commercial transaction frame is viewed.

The distributional view of semantics is that words' meanings can be found from the context words which have appeared together with the word in texts; this is a distributional hypothesis, or distributional semantics. The underlying idea of this hypothesis is *words which are similar in meaning occur in similar contexts*. This idea is familiarized by Rubenstein and Goodenough (1965a). Distributional semantics is the area where the dissertation has focused on. Distributional semantics finds the meaning of words by counting the co-occurrence of the word and the context word together in a large text. Then it constructs a high dimensional context vector to represent the word.

1.2 Motivation

Unsupervised machine learning has been widely used in Distributional Semantics. Distributional semantics is, basically, building an unsupervised framework for computational semantics. Computational semantics is finding a technique which computes meaning in human language and constructs a semantic representation for expressions of the language (Eijck and Unger, 2010). Besides building an unsupervised framework, unsupervised learning has also been a common approach for tasks in distributional semantics to build a model; Some examples are: to discover semantic similarity by similarity distance measures, to classify semantically related words by k-means method, etc.

In order to make further progress in the field of distributional semantics (especially, on building models for semantic relatedness tasks), extending the current unsupervised machine learning approaches to supervised learning approach needs to be investigated. The investigation is, specifically, to train a model on the supervised learning approach for semantic relatedness tasks such as semantic similarity, and compare the performance of the model with the unsupervised learning approach model. Supervised learning methods can play an important role in solving tasks in distributional semantics with better performance. Because, supervised learning uses label information of the training data to train the model, unlike unsupervised learning. Basically, the label information can improve the performance of the model for a specific task. For example, a model which classifies types of animals, such as wild and pet animals, will soon learn to identify animals after being trained on a dataset of animals that are properly labeled with their type. The first supervised learning approach study on semantic relatedness task is by Hagiwara (2008). Hagiwara (2008) has introduced the novel approach that is automatic synonym identification based on supervised machine learning and distributional features. We have also investigated this approach further with more task and different supervised learning methods.

In addition, the dissertation has introduced multi-relation matrix factorization (MRMF) as a supervised learning method in the distributional semantics field. MRMF provides an easy and effective way to integrate different sources of information. The performance of some tasks can be improved by integrating additional sources of information, especially when the words in the task are represented with additional sources of information. Since it is not easy and straightforward to integrate different sources of information in unsupervised learning, supervised learning is considered the right choice to be explored and to investigate MRMF performance in the field.

Multi-relation matrix factorization is basically a method for relation prediction in multi-relational domains using matrix factorization (Lippert et al., 2008). Matrix factorization is factorizing a matrix in two or more matrices. Matrix factorization is often applied for relation prediction and dimension reduction. Relation prediction is concerned with predicting unknown values of a relation from the given matrix of entities and observed relation values among entities. Dimension reduction is, basically, representing matrices by lower dimension. The well-known matrix factorization method is a singular value decomposition (SVD). The mathematical explanation of SVD can show us the intuition behind matrix factorization. Let's take matrix $\boldsymbol{M}$ which has size m x n. Then, $\boldsymbol{M}$ can be factorized in the form of $\boldsymbol{U}\Sigma\boldsymbol{V}^T$, where $\boldsymbol{U}$ is m $\times$ r right singular matrix, Σ is r $\times$ r rectangular diagonal matrix with non-negative real numbers, and $\boldsymbol{V}$ is n $\times$ r left singular matrix. r is the latent features size of the matrices. Latent reduces the dimension of a large number of directly observable features into a smaller set of features. When we multiply the decomposed matrices $\boldsymbol{U}$

and $\boldsymbol{V}$, it gives the original matrix; $X = UV^T$

Matrix factorization focuses on one relation type of two entities. However entities can have multiple relations. For example, students and classes have multiple relations such as grades, class dates. To model relation predictor of student and class, matrix factorization considers only one of their relation. However, it is possible to consider multiple relation domains to build a relation predictor using multi-relation matrix factorization. Multi-relation matrix factorization considers the multiple relations of entities. Multi-relation matrix factorization is the extension of single relation type of two entities to multi relation domain (Lippert et al., 2008). The aim of MRMF started on the performance improvement of relation prediction (Lippert et al., 2008). This method integrates multiple relation types from different sources of information easily and effectively using matrix factorization. It has been used intensively on social networks data, especially on recommended systems. Originally, MRMF studied user-movie recommendations and gene function prediction by Lippert et al. (2008). In the study, MovieLens[1] and yeast gene[2] datasets have used for user-movie recommendations and gene function prediction respectively. The datasets entity classes and their feature entities have been shown via the ER diagram in Figure 1.2. Figure 1.2 (a) shows entity classes *user* and *movie* and their feature entities *gender, age, occupation* and *genre*. Figure 1.2 (b) shows the simplified version of the gene data. As Lippert et al. (2008) stated, because of some logical constraints the sample of the gene is used from the whole dataset. The method MRMF tested against Matrix Factorization (MF) and Support Vector Machine (SVM) (only for gene function prediction). Then, MRMF outperformed MF and SVM.

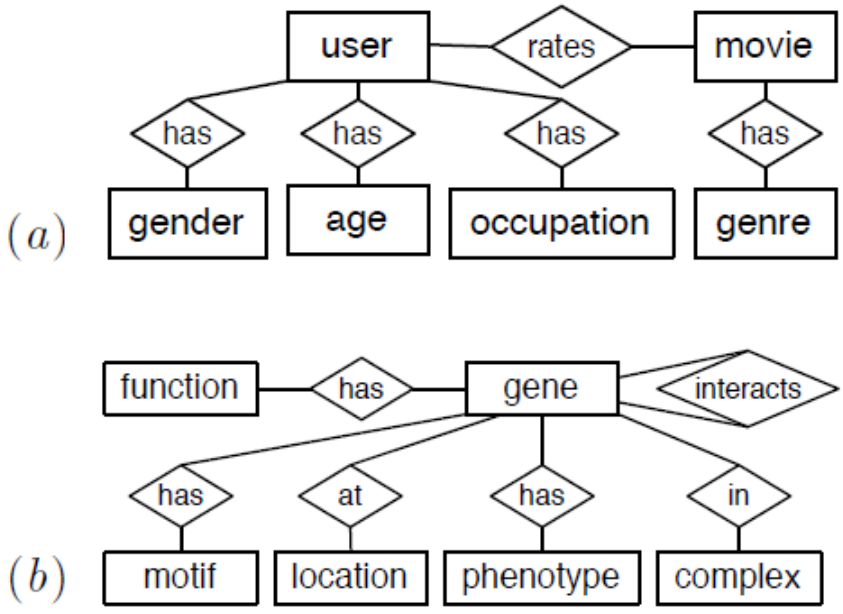

FIGURE 1.2: ER-diagrams showing (a) the MovieLens data and (b) an extract of the relations contained in the yeast gene data. Source: Lippert et al. (2008)

As a conclusion, the main motivation of this dissertation is attributed to the limited coverage and use of supervised machine learning approach to build a model

[1] https://movielens.org/
[2] http://mips.gsf.de/genre/proj/yeast/

for tasks in the distributional semantic field, unlike unsupervised machine learning. In addition, the dissertation studies MRMF performance in distributional semantic, and introduce the method in the field.

1.3 Research Problem

Applications and models of various tasks in distributional semantics such as predicting the semantic relation of a pair of words, categorizing semantically related words etc. have not fully reached their optimal level of accuracy yet. The performance and accuracy improvement in such applications and models is still very much possible. The unsupervised learning approach is one of the approaches that many tasks use intensively to build a model. However, these tasks are still far from reaching the maximum accuracy and performance level. Generally, in distributional semantics, many tasks have been tackled by an unsupervised learning approach. Thus, this dissertation is investigating a supervised learning approach to build a model for task, and study the model performance and accuracy improvement. Additionally, it explores the multi-relational matrix factorization on tasks in distributional semantics by considering different relations from different sources of information to build a model for a better performance.

This dissertation answers two general questions which are: 1. does supervised learning approach outperform the unsupervised learning approach to build a model for tasks in distributional semantics? 2. Can we use the multi-relational matrix factorization idea in distributional semantics to build a model which performs better for tasks? To answer these general questions, the following specific research questions on the specific tasks have been raised and answered.

- Which approach builds a model that improves the semantically related pair of words classification task, supervised or unsupervised learning?
- Which approach builds a model that improves the semantically related words categorization task, supervised or unsupervised learning?
- How is the performance of the tasks in distributional semantics using a model which is trained by multi-relational matrix factorization method?
- Which tasks' model in distributional semantics has improved the performance using multi-relational matrix factorization method?

1.4 Hypothesis

The dissertation has two main hypotheses. The first hypothesis is that models which have trained using supervised learning approach to solve tasks in distributional semantic outperform the unsupervised learning approach. For this hypothesis, the supervised machine learning approach performance has been investigated by conducting different tasks such as semantic similarity and semantic classification. The two tasks have explained briefly as follow: 1. The semantic similarity task is basically predicting whether pairs of words are semantically related or not. For example, semantically-related pairs of words are as follows: *car - automobile, moon - sun, noon - midday* and *gem - jewel*. 2. The semantic classification task is categorizing words to their semantic category. For example, the word *fruit* can be one semantic category, and the words which can be categorized under this category are fruits such as *orange, banana* and *apple*.

The second hypothesis is that the multi-relation matrix factorization method outperforms the tasks model in distributional semantics which have been trained by the well-known methods such as support vector machine (SVM) and logistic regression (LG). Like the first hypothesis, semantic similarity and semantic classification tasks have been considered in this hypothesis as well. In addition, evaluating a phrasal semantics task has also been included in this hypothesis. Evaluation of phrasal semantics task is evaluating the semantic similarity of a word and a short sequence of two words. For example, the word *girl* is similar with the sequence *young woman*. The aim of the second hypothesis is to introduce the MRMF method which is mainly known on social network analysis and recommender systems as an effective method, in distributional semantics.

There are large number of tasks that can be used to study the performance of supervised learning and multi-relation matrix factorization methods on distributional semantics. From the large number of tasks, semantic similarity and semantic classification tasks have been considered in this dissertation. Semantic similarity and semantic classification are some of the tasks that have been studied extensively in distributional semantics. Often, they have been using models which have been trained by an unsupervised learning approach.Thus, we can make a clear comparison of the supervised learning approach models performance with the unsupervised learning approach models performance. For this reason, these two tasks have been selected as the right candidates to use them in our studies. Apart from the two tasks, evaluating phrasal semantics task has included in our study for further investigation of multi-relation matrix factorization method.

Some studies like (Hagiwara, 2008; Shimizu et al., 2008; Turney, 2013a) have been recommending supervised leaning methods for semantic similarity tasks. Nevertheless, in this dissertation, their results have been used as a baseline to show the performance of the supervised learning methods that have been proposed in this study; Besides the unsupervised learning method result. For semantic classification task, the unsupervised learning approach method result is used as a baseline. As far as we know, the latest state of the art model of semantic classification task has trained on unsupervised learning approach method. Each of our study state of the art has explained in detail in Chapter 2.

1.5 Contribution and Structure

This section presents the contribution of this dissertation and its structure.

Chapter 2: Distributional Semantics

This chapter presents what distributional semantics is. It explains the distributional semantics word representation and which procedures the dissertation followed to represent the words from the experiment dataset by the high dimensional vector. It also explains briefly each contribution of the dissertation or project with their related works.

Chapter 3: Semantic Similarity

In this chapter, one of the dissertation contributions which is published as *Learning Thesaurus Relations from Distributional Features* (Aga et al. (2016b)) is presented. This study presents a supervised method which outperforms the state of the art method. Then, the method is proposed for applications like maintaining and constructing thesaurus.

Chapter 4: Semantic Classification

The work which is published as *Integrating Distributional and Lexical Information for Semantic Classification using MRMF* (Aga et al. (2016a)) is presented in this chapter. The study introduces a supervised learning method for the first time as accurate method for categorizing semantically-related words by outperforming the state of the art unsupervised learning methods. Additionally, it introduces multi-relation matrix factorization for the first time in distributional semantics by outperforming the selected supervised learning methods for the task which outperformed the unsupervised learning methods.

Chapter 5: Identification of Semantic Relation

This chapter explains the work that is published as *CogALex-V Shared Task: HsH-Supervised – Supervised Similarity Learning using entry wise product of context vectors* (Tsegaye and Wartena (2016)) . This study is basically a shared task challenge. The task is to recommend a method which semantically classifies related pairs of words. For this challenge, the method which is proposed in Chapter 3 has been studied on the given shared task. In addition, the multi-relational matrix factorization method performance has also been investigated.

Chapter 6: Evaluating Phrasal Semantics

This chapter explains the multi-relation matrix factorization method performance study on SemVal-2013 shared Task 5a challenge. The challenge is evaluating phrasal words. For this challenge, the state of the art proposes a supervised learning method. However, the multi-relation matrix factorization method positive performance on other tasks (like semantic classification) has been motivated this study.

Chapter 7: Applications

This chapter discusses the three applications that are published as *Constructing concept clouds from company websites* (Aga and Wartena (2015)), *Automatic Recognition and Disambiguation of Library of Congress Subject Headings* (Aga, Wartena, and Franke-Maier (2016)) and *Automatic Identification of Synonym Relations in the Dutch Parliament Thesaurus* (Aga et al. (2017)).

Chapter 2

Distributional Semantics

2.1 Introduction

The meaning of a word can be determined by using context. Context refers to the words in sentences around the target word. Target word is the unfamiliar or unknown word that we are trying to find the meaning using the words in a sentence as a clue. The meaning of a word can often be found from clues in the surrounding context. Often, what comes before and after the target word can reveal the meaning of the word. The surrounding words can give helpful context clues about the meaning and structure of the target word.

There are several methods or context clue types for using contexts to figure out the meaning of a word. We can see some of them briefly with an example. One of the clue types is to see if the definition of the word is in the sentence. This method is more like restating the target word. The target word may have been defined sufficiently within the sentence. For example, *The manager wanted a weekly* ***inspection****, which is a methodical examination of all the equipment*. The other type is using listed examples in the sentence where the target word is mentioned. Words like *including*, *such as*, and *for example*, point out example clues. For example: ***Piscatorial*** *creatures, such as flounder, salmon, and trout, live in the coldest parts of the ocean*. Autonomy or contrast is the other clue type. This clue type contrasts the meaning of the target word with the meaning of a familiar or known term. For example: *The picture of the landscape is* ***picturesque*** *but the one of the old houses is ugly.* As we have seen some of the context clue types, we can say that the meaning of a word is the set of contexts that occur in texts. Context clue is a source of information about a word that helps to understand the word. The clue offers insight, either directly or indirectly, into the word's meaning. Besides written texts, the context of a word can be found within audios and videos.

Distributional semantics is mainly about finding the meaning of a word using the context information from the texts where the word has appeared. Distributional semantics studies theories and methods, that extract semantic meaning/information from the way words behave in a large and structured set of texts which is called text corpora. Then, it quantifies and categorizes semantic similarities between the words based on their distributional properties in large samples of language data. Distributional semantics follows the idea of words with similar meanings that will tend to occur in similar contexts.

In distributional semantics, words are represented by context features, and usually it is the co-occurrence numbers or the point-wise mutual information between each word and context word. The point-wise mutual information (PMI) is a measure of association between words. It measures how often two events x and y occur, compared with what we would expect if they were independent. Mathematically it has shown in Equation 2.1:

$$PMI(x,y) = \log \frac{p(x,y)}{p(x)p(y)} \tag{2.1}$$

When we apply the point-wise mutual information on the co-occurrence of target and context words, event x becomes the target word w and event y becomes context word c. The target and context words point-wise mutual information has shown in Equation 2.2.

$$PMI(w,c) = \log \frac{p(w,c)}{p(w)p(c)} \tag{2.2}$$

Distributional semantics favors the use of linear algebra as a computational tool and representational framework. The basic approach is to collect distributional information in high-dimensional vector space. The structure of this space is provided by considering the contexts in which words occur in large corpora of text. Word co-occurrence statistics can provide a natural basis for words semantic representation.

A number of quotes have been given for distributional semantics; some examples from the very common sayings are, *You shall know a word by the company it keeps* by Firth (1957), *Difference of meaning correlates with difference of distribution* by Harris (1954) and *Words that occur in similar contexts tend to have similar meanings* by Turney and Pantel (2010). However, the main question is on how to operationalize this idea and use it in different tasks such as semantic classification and semantic similarity. Specifically, the question is how do we represent words by a vector space and use the representation for tasks in distributional semantics. Distributional models of meaning or the vector space of word representation are generally based on a co-occurrence matrix. The matrix has the information of how often the word and context word co-occur. Let's see one of the very common distributional semantic co-occurrence matrix examples to clear up the idea of word representation. In Table 2.1, the first row contains the context words, and the first column contains target words. The contextual words are the distributional information which has been extracted from the corpus which contains the target words as well. The cells in the table contain the number of times that the target word and context words co-occur in the corpus. The context words are *get, see, use hear, eat* and *kill*. If we see the word *cat* row as an example, it contains $[52, 58, 6, 26]$. This row is the context words vector representation of the word *cat*. This vector shows that the word *cat* and *get* co-occur in the corpus 52 times, the word *cat* and *see* co-occur in the corpus 58 times, the word *cat* and *use* co-occur in the corpus 4 times, and so on. This matrix has constructed with a simple frequency of the co-occurrence of two words (target word and context word). Thus, to have the best vector representation, more parameters have to be analyzed and investigated. The distributed representation of words has to be learned primarily with respect to the parameters such as context type, context windows, frequency weighting and dimension reduction. For instance, frequency is not the best measure of the two words' relation. As Jurafsky and Martin (2000) stated in their book, one of the problems is that raw frequency is very skewed and not very discriminative. In addition, some researchers (Bullinaria and Levy, 2012; Kiela and Clark, 2014) showed that applying weighting measures like point-wise mutual information is recommended, instead of raw frequency for a better representation of words with respect to other parameters. Each parameter has explained in detail in Section 2.5.1.

	get	see	use	hear	eat	kill
knife	51	20	84	0	3	0
cat	52	58	4	4	6	26
dog	115	83	10	42	33	17
boat	59	39	23	4	0	0
cup	98	14	6	2	1	0
pig	12	17	3	2	9	27
banana	11	2	2	0	18	0

TABLE 2.1: Co-occurrence matrix

Words with a similar meaning have similar vectors of context features. In other words, semantically similar words occur in similar contexts. Rubenstein and Goodenough (1965b) showed this idea via their semantic similarity study pair of words using the context vector of the words of the pair. In Rubenstein and Goodenough (1965b) study, a pair of words with common context words showed a positive relation. Especially, a pair of words with a large amount of context words overlapping showed high relation in their study. Thus, words can easily be compared for similarity in the vector space using any distance similarity measures such as the cosine of the angle between two vectors. The cosine measure is computed as the normalized dot product of two vectors. For example, let us see the semantic similarity of the words *cat* and *dog* using cosine distance measure. We can take the vector space representation of the two words from Table 2.1. The *cat* vector representation is *[52,58,4,4,6,26]*, and the *dog* vector representation is *[115,83,10,42,33,17]*. Then, we can use Equation 2.3 to compute their similarity distance cosine. In Equation 2.3, A and B are variables for the *cat* and *dog* vector representation respectively. A_i and B_i are components of vector A and B respectively. In cosine, when two words are the same or related, the cosine result is 1 or close to 1; otherwise 0 or close to 0. The relatedness of the example words has been computed using the cosine Equation 2.3. Then it gave 0.07. This means that the words *cat* and *dog* are not semantically related. . Basically, *cat* and *dog* are related or not related depending on the context. As Table 2.1 shows, the context words of the two words are *get, see, use, hear, eat* and *kill*. Based on this context, *cat* and *dog* are not related. Because this context focuses more on the behavior of the two animals. Obviously, the behavior of dog and cat are not that related. However, we can not conclude that they are related or not with the given vector representation. This is because the vector representation of these words is not the best vector representation which has been constructed with different parameters and parameter values selection.

$$cosine(A,B) = \frac{A.B}{\parallel A \parallel \parallel B \parallel} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{2.3}$$

In the dissertation, the first and main part is to represent each word of the dissertation projects' datasets by semantic vector representation. The vector representation of each word has been computed with respect to different parameters and parameter values selection to get the best representation. Each word vector representation has been used on different supervised machine learning techniques to solve different tasks in distributional semantics. The parameters and parameter values selections have been explained in Section 2.5.1. The main tasks that we have covered in this

dissertation are semantics similarities and semantic classification. In addition, some distributional semantic applications which use the words' representation have also been introduced. These tasks and applications have explained briefly in the following sections.

2.2 Distributional Similarity

Distributional similarity has been widely studied to solve many different tasks related to the meaning of words. Some of the tasks are:

- Word sense disambiguation which is identifying what sense or meaning that a word has in a sentence, when a word has multiple meanings
- Paraphrasing, which is expressing the meaning of something written or spoken using different words
- Spelling correction
- Query expansion, which is reformulating a given query to improve retrieval performance in information retrieval operations like search engines (e.g. Google).

Later, distributional similarity became an established method to find similar words. As Harris (1954) explained in his distributional hypothesis, the degree of semantic similarity between two linguistic expressions A and B is a function of the similarity of the linguistic contexts in which A and B can appear. This means that two words will have high distributional similarity if their surrounding contexts are similar.

The similarity of words can be computed by comparing their feature vectors. Thus, it is possible to predict whether two words are semantically related or not with respect to their semantic relation (Synonym, antonym, hyponym, etc.). In distributional similarity, similarity measures like Cosine, Euclidean distance and Jaccard's coefficient are well-known methods to find semantic similarity between words (like a pair of words). Some of the similarity measure mathematical equations are:

- Euclidean distance measures the geometric distance. The metric is established in one dimension by fixing two points on a line between the two vectors

$$Eculidean(A, B) = \sqrt{\sum_{i=1}^{n}(A_i - B_i)^2} \tag{2.4}$$

- Jaccard assesses the amount of weighted overlap between features.

$$Jaccard(A, B) = \frac{\sum_{i=1}^{n} min(A_i, B_i)}{\sum_{i=1}^{n} max(A_i, B_i)} \tag{2.5}$$

In Equation 2.4 and 2.5, A and B are the two words of the pair vector representation variables. A_i and B_i are components of vector A and B respectively. The cosine measure has explained in the previous section with Equation 2.3.

Many research pieces have been comparing these and many more semantic similarity measures for different reasons. For example, Lee (2000) compared different semantic similarity measures for the purpose of automatic extraction of semantic

similarity measures from corpora. These similarity distance measures are unsupervised methods to classify semantically-related pair of words.

In this dissertation, a supervised method has been introduced to classify semantically-related pair of words. The supervised method performance has been compared with the unsupervised semantic similarity measures. In our study, the cosine has been shown a better performance from the list of unsupervised semantic similarity measures such as Euclidean distance and Jaccard. Thus, the cosine has been selected as one of the unsupervised semantic similarity measures to be used as a baseline for our supervised method. The supervised method relies on a support vector machine algorithm and the distributional vector representation of the pair of words. The pair of words has been using simple mathematical equations like *addition* and *multiplication* in order to construct the vector representation.

Meaning representation using addition and multiplication operations has introduced by Mitchell and Lapata (2008). Mitchell and Lapata (2008) proposed this framework to represent phrases and sentences meaning by vector space. In their study, they have introduced models based on addition and multiplication which evaluate sentences similarity task. They used *Noun* and *Verb* as target words from a sentence to apply the addition and multiplication operations. For example, a sentence like *Horse runs*. The models estimate the similarity of the sentences by following unsupervised methods such as the cosine similarity distance measure. They have represented each word by a context vector from a British National Corpus (BNC)[1]. They have experimented variety of dimensions (ranging from 50 to 500,000) to represent each word. Mitchell and Lapata (2008) best result obtained with a model using a context window of five words on either side of the target word, the cosine measure, and 2,000 vectors. Let us see one of (Mitchell and Lapata, 2008)'s examples which show us how the two operations have been used to represent sentences' meaning. In the example, each word has represented by five context words (*animal, stable, village, gallop, jokey*). *Horse* is represented by [0, 6, 2, 10, 4], and *run* by [1, 8, 4, 4, 0]. The vectors value is the co-occurrence of the target words (horse and run) with the context words. Since they have used, lemmatized version of the corpus, the example word *runs* becomes *run*. The addition operation represents the sentence by **house + run** = [1,14,6,14,4] vector space, and the multiplication operation represents by **house** · **run** = [0,48,8,40,0] vector space.

Besides the distributional information, a pair of words can be represented by a small number of features which have the distributional proprieties of the pair of words. Turney (2013a) proposed an approach which aggregates different types of features to represent the pair of words and predicts their semantic relation. These different types of features are a combination of different functions and are also based on frequencies in a large corpus. These features either represent properties of one or both of words of the pair, or a property of the pair, for instance, the point-wise mutual information of these words. Since the Turney (2013a) study related with ours, their study approach has also been considered in our study to construct the pair of words vector representation. Then, a model has been trained on SVM by taking the pair of words vector representation as an input. We called this model *SuperSim*, as the study has shown that our model outperformed the *SuperSim* model as well. The use of such a set of features to represent a pair of words is followed by Santus et al. (2016) as well. Santus et al. (2016) presented a ROOT9 supervised system for the classification of hypernyms, co-hyponyms, and a random pair of words. The system

[1] http://www.natcorp.ox.ac.uk/corpus/index.xml

relies on unsupervised corpus base features to represent the pair of words. The feature set designed to identify the distributional properties of the pair of words like the co-occurrence of the pair of words and frequency of each word of the pair.

The mathematical equations simply take the context vector of each word of the pair and apply the equation element-wise in order to represent the pairs. *SuperSim*, as well, aggregates different functions to represent the meaning of pairs. In our study, these pairs vector representation has been used as an input, directly, for the SVM algorithm to train a model which can classify semantically-related pairs of words. The model has been outperformed the cosine similarity measure. This result has been shown on six different semantically related and not a related pair of words datasets which have constructed from thesauri and state of the art dataset. The datasets, the pair of words vector representation and training the model have explained in detail in Chapter 3.

One of the studies which is closely related to our work is the Weeds et al. (2014) study. Weeds et al. (2014) have also studied a supervised approach on the tasks to predict hypernyms and co-hyponyms semantic relations between a pair of words given their distributional vector. In Weeds et al. (2014)'s study, they have also used the mathematical operations such as addition and multiplication in order to represent a pair of words and use the vector representation as an input for SVM to build a classifier. They have compared their results with unsupervised approach methods such as cosine. Then, for hypernyms relation prediction task, the SVM methods, generally, outperformed the unsupervised methods outperform the unsupervised methods. However, for the co-hyponyms semantic relation, the unsupervised method cosine has outperformed the supervised methods. In our study, as well, we use the mathematical operations to represent the pairs, and use representation as an input to SVM to train a classifier which classifies pairs that have a co-hyponyms semantic relation. However, the procedure that they have followed to represent each word by context vector before the pair's representation is quite different. They have collected the distributional information from Wikipedia. Words which have occurred 100 or more times have collected as features. Then, PPMI has applied on the feature values after tagging and lemmatising the words. The way that we have been representing words meaning have explained in detail in Section 2.5. In addition, we use unit vector on the context vector before applying the mathematical operations on the context vectors to represent the pairs. Finally, our supervised approach has outperformed the unsupervised approach to classify pairs which have co-hypernym relation. In our study, further, we have investigated different types of pairs by their co-hypernym relations distance such as broad co-hypernym relation and tight co-hypernym relation. These different types of pair datasets have been constructed from thesauri. This and more have been explained in detail in Chapter 3.

The other work which is closely related to our work is the research on metric learning for distributional similarity from Shimizu et al. (2008) and Hagiwara (2008). Shimizu et al. (2008) used a learned Mahalanobis distance to rank pairs of related and unrelated words. In order to make the learning computationally feasible they reduced the number of context features massively by selecting the most promising features. Then,they formalized the Mahalanobis method as follows. Given points x_i, $x_j \in R^d$. R^d is a sparse vector which represents the contexts of the target word. The (squared) Mahalanobis distance parameterized between the points by a positive definite matrix Q as follows: $d_Q(x_i, x_j) = (x_i - x_j)^T Q(x_i - x_j)$. Then, they have stated that the Mahalanobis distance between the vectors of two synonymous words must be smaller than a given upper bound, i.e., $d_Q(x_i, x_j) \leq u$ for a relatively small

value of u; and two words are dissimilar if $d_Q(x_i, x_j) \geq l$ for sufficiently large l. Their main objective is to obtain the positive definite matrix Q that parameterizes the Mahalanobis distance. In their study they use the Euclidean distance to obtain synonyms quite well. Therefore, they defined the positive definite matrix Q of the Mahalanobis distance to be close to the identity matrix I. To do this, they started to use a simple bijection (up to a scaling function) from the set of Mahalanobis distances to the set of equal mean multivariate Gaussian distributions. Gaussian is a very common continuous probability distribution. Normal distributions are important in statistics and more to represent real-valued random variables whose distributions are not known. The corresponding Gaussian is shown in Equation 2.6; Where Z is the normalizing factor. Then they measured the distance between two Mahalanobis distances using the Kullback-Leibler (KL) divergence of two Gaussians as shown in Equation 2.8. Their study optimization problem is shown in Equation 2.7 with the given pairs of similar points S and pairs of dissimilar points D. They applied metric learning to automatic synonym acquisition. Here as well, we followed a different approach to train a model which classifies the pairs' relation.

$$p(x; Q) = \frac{1}{Z} exp(-\frac{1}{2} d_Q(x, \mu)) \tag{2.6}$$

$$KL(p(x; I) \parallel p(x; Q)) = \int p(x, I) log(\frac{p(x; I)}{p(x; Q)}) dx \tag{2.7}$$

$$\begin{aligned} \text{minimize}_Q \quad & KL(p(x; I) \parallel p(x; Q)) \\ \text{subject to} \quad & d_Q(x_i, x_j) \leq u(i, j) \in S \\ & d_Q(x_i, x_j) \geq u(i, j) \in D \end{aligned} \tag{2.8}$$

In the approach of Hagiwara (2008), feature selection is not necessary as one of the constructed features to represent each pair of words. For the pair of words, they defined distributional features $f^D{}_j(A, B)$ using the co-occurrence as shown in Equation 2.9. The feature value is determined to represents the degree of commonality of the context shared by the word pair. Then they adopted pointwise total correlation as shown in Equation 2.10. The PMI formula is shown in Equation 2.2. Subsequently, SVM is used to learn that which pairs are pairs of synonyms and which are not. We have followed a similar approach to classify the pairs, but we use a different approach to construct the single words features and the pairs vector representation.

$$f^D{}_j(A, B) = log \frac{P(A, B, c_j)}{P(A)P(B)P(c_j)} \tag{2.9}$$

$$f^D{}_j(A, B) = PMI(A, c_j) + PMI(B, c_j) \tag{2.10}$$

One of the practical applications of distributional similarity that is often mentioned, is automatic updating and extension of a thesaurus with new terminology (Crouch, 1990; Curran and Moens, 2002). Crouch (1990) compared the early and current approaches to automatic thesaurus construction. Then, Crouch (1990) described an approach to the automatic generation of global thesauri. Curran and Moens (2002) evaluated existing and new similarity metrics for thesaurus extraction.

Then, they proposed an approximation algorithm which reduces the time complexity and execution time of thesaurus extraction. However, we are not aware of any attempt to extend large existing thesauri with new terms or concepts based on distributional similarity. The semantic relation, that we have considered, is relatedness of terms in thesauri for intellectual document classification. Thus our findings can directly be applied for the maintenance and extension of such thesauri. To the best of our knowledge this relation was not considered before in the field of distributional semantics. Our distributional similarity project has been explained in detail in Chapter 3.

2.3 Distributional Semantic Classification

Semantic classification is one of the distributional semantics tasks. The definition of semantic classification in the context of this dissertation is a task which categorizes words by their semantic category. For example, let us take a list of words: *chair*, *table*, *banana* and *apple*; and, two categories: *furniture* and *fruits*. Then when we categorize the list of words to their semantic category using semantic classification method, the words *banana* and *apple* can be categorized under *fruits*, while the words *chair* and *table* can be categorized under the *furniture* category. There is some number of unsupervised learning methods such as k-means, to categorize words to their semantic category. Thus, this has motivated our study on semantic classification task to find a better method which categorizes semantically-related words to their right semantic category by following a supervised learning approach.

Semantic classification of words using distributional features is usually based on the semantic similarity of the words. Basically, the semantic similarity measures the similarity distance between the list of words. Then, the words can be clustered based on their similarity distance. Words with small distances can be categorized under the same category. This is the unsupervised approach of the semantic classification task. In this dissertation, we have been showing that classifiers which have been trained on supervised methods outperform the unsupervised learning approach. These classifiers have been trained by using the distributional features directly as an input to the supervised learning methods. In order to train the classifiers, three different supervised learning algorithms have been used. These are Logistic regression (LR), SVM, and Multi-Relational Matrix factorization (MRMF). The classifiers which have been trained on these three algorithms have outperformed the unsupervised learning classifiers. However, to improve their performance for higher accuracy, lexical information has been integrated with the distributional features information. The direct definition of lexical is the meaning of the term in common usage, this also known as dictionary or vocabulary. The classifier which has been trained on MRMF has improved the performance after integrating the lexical information. However, the classifiers which have been trained on LR and SVM have not shown performance improvement. But still, their performance is better than the unsupervised learning classifiers.

In distributional semantics, MRMF is a novel approach for semantic classification task. MRMF has a nature of being easily extended with more matrices which contain more information from different sources on the same problem. In addition to introducing supervised classifiers for the semantic classification task, the dissertation demonstrates the effectiveness of the novel approach on two ways: 1. by using only distributional information as an input 2. by integrating lexical information with distributional information, and use the integration as an input to train a

classifier. Thus, we have shown that MRMF provides an interesting approach for building semantic classifiers that (1) gives better results when compared to unsupervised learning approach which is based on semantic similarity, (2) gives the same results as SVM and better than LR when the input data is only distributional information (3) can naturally be extended with more sources of information in order to improve the results. The lexical information that has been integrated with the distributional information has been taken from WordNet[2].

Matrix factorization has been used in distributional semantics, e.g. by Giesbrecht (2010) and Cruys, Poibeau, and Korhonen (2013) in order to reduce the size of the feature space, but not directly for predicting missing values or for classification. We are not aware of any work using matrix factorization for arranging the classification of words into semantic categories.

The integration of distributional and lexical information is an obvious way to go and is also used in a large number of studies like (Finkelstein et al., 2001) and (Yih and Qazvinian, 2012). Usually, a weighted average of similarities based on different types of information is used. E.g. Finkelstein et al. (2001) used distributional features (occurrence frequencies of words in various domains) and the cosine of these feature vectors as a distributional similarity measure. This measure is combined linearly with a WordNet-based similarity measure. Yih and Qazvinian, 2012 use different similarity methods, like corpus-based and web-based distributional similarity for binary classification tasks (synonymous or not-synonymous). They also used WordNet similarity. For this, they represent a word as a vector in a Synset-space. Thus, the vector indicates which synsets a word belongs. Finally, they aggregated the various similarities by taking the average cosine similarity. Camacho-Collados, Pilehvar, and Navigli (2015) combined distributional similarity of words based on their occurrence in Wikipedia with a WordNet based similarity measure. They also combined the similarities from both sources by computing the average. Pennacchiotti et al. (2008) furthermore investigate the contribution of distributional models and their combination with Wordnet. They use the simple back-off model to combine distributional similarity and WordNet-based similarity.

Classification of words into different semantic categories has studied by Pekar, Krkoska, and Staab (2004), who use a k-Nearest Neighbor classifier and investigate different feature weighting schemes and distance measures. They studied several feature weighting methods in application to automatic word classification. Their focus was on the differences between those weighting methods. The methods are odds ratio, gain ratio and mutual information. For the study, they investigated some distance measures like Jensen Shannon, Jaccard and Cosine. Then they selected Jensen Shannon as the best distance measure for the task. In their study, they found that discriminative and characteristic weighting procedures are able to identify different kinds of features which are useful for learning a classifier. This enhances the classification's accuracy. Both Bullinaria and Levy (2012) and Keith, Westbury, and Goldman (2015) used the nearest centroid classifier for the word classification task on the dataset, that we have also used this in our study. In their study, first, they computed the average context vector of each category by excluding the test set word context vector from the category. Then the test set word distance computed with the average context vector of each category. The smallest distance defines the category of the test word. Bullinaria and Levy (2012) used this procedure on ten-cross-validation. However, Keith, Westbury, and Goldman (2015) reported only the result for one arbitrary split into test and training sets. Thus, their results cannot be compared directly to

[2] https://wordnet.princeton.edu/

our results and to Bullinaria and Levy (2012)'s results as well. The dissertation distributional semantic classification project has been explained in more detail in Chapter 4.

2.4 Distributional Semantic Application

Distributional semantic models were successfully applied for a number of tasks like: finding semantic similarity between words and multi-word expression, word clustering based on semantic similarity, automatic creation of thesauri and bilingual dictionaries, lexical ambiguity resolution, expanding search requests using synonyms and associations, defining the topic of a document, document clustering for information retrieval, data mining and named entities recognition, creating semantic maps of different subject domains, and many more. The dissertation presents three applications which have been constructed using distributional information. These are: 1. Concept clouds from German company websites, 2. Automatic Recognition/ Disambiguation of Library of Congress Subject Headings, 3. Automatic Identification of Synonym Relations in the Dutch Parliament Thesaurus.

The first application is related to word cloud based on semantic similarity. Word clouds are used for the visual representation of texts. The font size and color of the word demonstrate the importance of the word. The position of the word in the cloud can be arbitrary or reflect. This shows the relation of the word with the other words. In this dissertation, a tool has been presented which generates concept clouds from German company websites. The main idea of the visualization is to show the overall work and main interests of companies in a detailed cloud information based using their own web page.

The second application is assigning Library of Congress Subject Headings[3] (LCSH) terms automatically for indexing. LCSH is a thesaurus of subject headings which is maintained by the United States Library of Congress, for use in bibliographic records. The terms have been assigned automatically by extracting them from the abstract of records which have been manually annotated with LCSH. The third application is to find synonyms term for the Dutch parliament thesaurus automatically.

The three applications have explained in detail with their related work in Chapter 7.

2.5 Word Representations

Most tasks in natural language processing have been solved by representing words with a contextual information vector. Words have ambiguous meanings, finding the good vector representation is one of the natural language processing challenge and main work. One of the known and usual ways of finding the meaning of a word is reading the words which have appeared next to the target word. As Firth (1957) said *You shall know a word by the company it keeps*. The words that appear next to the main word can tell the context of the target word as it has explained at the beginning of this chapter. For example, let us see the word *beat* which has ambiguous meanings. When we use the word *beat* in a sentence, we have to see the context words to find what the word *beat* means in the sentence because beat has multiple meanings. Some of the meanings of the word *beat* can be musical rhythms,

[3] http://id.loc.gov/authorities/subjects.html

hitting someone, a heartbeat, etc. Therefore, we have to see the contextual words of the target word to understand what meaning it has in the sentence.

To represent a word by a context vector, the context words and the target word co-occurrence is counted from a large number of sentences (corpus). The context words of the target word are considered as features, and the co-occurrence of the target word and the context word in a corpus creates context vector for the target word. Then, the context vector represents the target word. This way of representing a word is a simple word-word co-occurrence. Table 2.1 is one example of word-word co-occurrence.

The quality of word representation is dependent on the parameters and parameter values that we select in order to construct the context vector. Some of the parameters that have to be examined for a good word representation, are: the window size which defines the range of neighboring words that we need to consider as context words, the frequency range of words in a corpus which guarantee a good context words selection, the size of the corpus to have large enough context words, the type of normalization method to adjust the context vector values to a notionally common scale, and many more. Bullinaria and Levy (2007) studied word representation and set out a general framework for generating the semantic vectors that represent words. They presented the framework from a systematic series of computational experiments. They examined the framework for a range of semantic tasks such as semantic classification and TOEFL (Test of English as a Foreign Language) to estimate the semantic validity of their representation. The framework has been designed to examine how different statistical collection details affect the performance of the resultant co-occurrence vectors as semantic representations. Some of the main statistical computation that they studied are the corpus size, context window size, and the vector normalization method. Each test ran on similarity distance measures (such as Euclidean and Cosine distance) on the space of semantic vectors.

In the Bullinaria and Levy (2012) study, they considered two corpora, BNC and ukWaC corpora. BNC[4] is a 100 million word collection of samples of written and spoken British English language from a wide range of sources from the later 20th century part. ukWaC is also a British English corpus. ukWaC[5] contains 2 billion words from the *.uk* domain using medium-frequency words from the BNC as seeds words. For a better performance, they cleaned stop words from the corpora. Example of stop words: *is, are, he, she, they*. In addition, they lemmatized and stemmed the words in the corpora. To study, the corpus size effect on semantic representation, they split each corpus into N equally sized disjoint subsets (N = 1, 2, 4, 8 for the BNC, N = 1, 2, 3, 6, 12, 24 for the ukWaC) (Bullinaria and Levy, 2012). Then, they showed results improvement with the corpus size increasing. In addition, they showed that using the small window sizes, which are one word each side of the target word, gives the best results. For the semantic vectors, they considered PMI as an alternative to the raw probability. PMI can have negative or positive values. If it is zero, the target t and context words c are independent. PMI maximizes when t and c are perfectly associated. In Bullinaria and Levy (2012) study, they set all the negative PMI values to zero. They compared vectors of Positive Point-wise Mutual Information (PPMI) with PMI and raw probabilities of the co-occurrence. Then, PPMI gave the best performing semantic representations in all their semantic tasks with respect to very small context windows, cosine distance measure and large corpus size. Equation 2.11 shows PPMI.

[4] http://www.natcorp.ox.ac.uk/corpus/index.xml

[5] https://www.sketchengine.eu/ukwac-british-english-corpus/

<s>		
Hooligans	NNS	hooligan
,	,	
unbridled	JJ	unbridled
passion	NN	passion
-	:	-
and	CC	and
no	DT	no
executive	JJ	executive
boxes	NNS	box
.	SENT	.
</s>		

TABLE 2.2: ukWaC sample data. 1^{st} column: The original text; 2^{nd} column: part-of-speech tag of the text; 3^{rd} column: lemma of the text

$$ppmi(c,t) = \max\left(\log\frac{p(c|t)}{p(c)}, 0\right). \tag{2.11}$$

In this dissertation, words have been represented by distributional representation. The distribution representation has been structured by following Bullinaria and Levy (2007) and Bullinaria and Levy (2012) framework. This representation has been used as an input for many of the projects in the dissertation. Bullinaria and Levy (2007) have started the study on the parameters window size, similarity distance measures, frequency range of context words, corpus size and the vector normalization methods. Then, they have extended their study and investigated the use of three further factors which are stop-lists, word stemming, and dimensionality reduction using SVD (Bullinaria and Levy, 2012). In our study, we have investigated dimensionality reduction using SVD for our PPMI matrix. However, the dimention reduction parameter did not help us to improve the quality of the vector representation. Therefore, we have excluded this parameter from the framework. The following Subsection 2.5.1 explains how the distributional features have been constructed in the dissertation.

The recent word representation is word embedding. Word embedding and distributional representation share the same goal, that of representing words. However, they both have their own differences. The following Subsection 2.5.2 explains word embedding and its difference with distributional representation.

2.5.1 Distributional Representation

This section explains the distributional vector representation of each word in this dissertation. To construct the vector representation for each word, two corpora (one for English and one for German words) have been used. For the English words, ukWaC, the British English corpus from the *.uk* domain has been used. ukWaC is one of the biggest corpora which is used for the success of good results in a number of different tasks in Bullinaria and Levy (2012) study. Therefore, this corpus has been considered in our studies as well. The words in the corpus tagged and lemmatized using TreeTagger. TreeTagger[6] is a tool for tagging text with their part-of-speech and lemma words. The commonly listed English parts-of-speech are *noun, verb, adjective, adverb, pronoun, preposition*. This tool has been successfully used for many more languages such as German, French, Italian, Danish, Dutch and Spanish. Sample data

[6] http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/

<s>		
Mit	APPR	mit
dem	ART	d
vorliegenden	ADJA	vorliegend
Buch	NN	Buch
wird	VAFIN	werden
,	,	,
zwei	CARD	zwei
Jahre	NN	Jahr
nach	APPR	nach
dem	ART	d
Erscheinen	NN	Erscheinen
von	APPR	von
Werner	NE	Werner
Arnolds	NE	Arnold
Lehrbuch	NN	Lehrbuch
des	AR	T d
Neuwestaramäischen	NN	Neuwestaramäischen
,	,	,
Unter-richtsmaterial	NN	Unter-richtsmaterial
für	APPR	für
eine	ART	ein
weitere	ADJA	weit
neuaramäische	ADJA	neuaramäische
Sprache	NN	Sprache
bereitgestellt	VVPP	bereitstellen
.	.	.
</s>		

TABLE 2.3: deWaC sample data.1^{st} column:The original text; 2^{nd} column: part-of-speech tag of the text; 3^{rd} column: lemma of the text

of ukWaC corpus with the part-of-speech and lemma words has shown in Table 2.2. The corpus saved the data in the *xml* format. Table 2.2 shows <*s*> and </*s*> which indicate the start of a sentence and the closing of a sentence, respectively. For the German words, deWaC corpus has been used. The DeWaC corpus [7] made up of texts collected from the internet in *.de* domain. The corpus structured (like ukWaC) with part-of-speech tagging and lemma using TreeTagger and stored in xml format. The sample data of the corpus has shown in Table 2.3.

Before the context vectors' construction, the corpora structured by cleaning the stop words, extracting only the lemma words, and changing the format to a document (*.txt* file) from xml. Structuring the corpora helps to decrease the size of the data by half, this helps to decrease the computation time. For example, Table 2.2 sample data has structured as follow *hooligan unbridled passion executive box*. The idea of cleaning stop words and considering only lemma words has taken from Bullinaria and Levy (2012) vector representation framework. When we construct the vector representation, some number of parameters and parameter values have been evaluated. This evaluation has been done on parameters and parameter values that turned out to yield the best results in a number of different tasks in studies by Bullinaria and Levy (2007) and Bullinaria and Levy (2012).

The parameters which have been evaluated for the vector representation are window size, features size, and the vectors normalization methods. First, we have determined which words have to be used as context words, i.e. with which context words

[7] https://www.sketchengine.eu/dewac-german-corpus/

that the target word co-occurrence statistics have to be computed. Thus, the window size of two words has been determined to consider one word from the right and one word from left as context words, while respecting sentence boundaries. Bullinaria and Levy (2007) and Bullinaria and Levy (2012) showed that smaller windows yield better results if the training corpus is large enough. Before the co-occurrence computation, all stop words have been removed when structuring the corpus. Therefore, syntactic relations have not been used in our studies.

Next, the right frequency range of words from the corpus has been used to select context features. As per the Bullinaria and Levy (2007) and Bullinaria and Levy (2012) findings, mid-frequency words are most effective. After some preliminary experiments, we have also found that including all words in the mid-frequency range, especially from $4 \cdot 10^3$ to $1 \cdot 10^6$ in the ukWaC Corpus as a context feature, is a good compromise between optimal results and acceptable storage and computing efforts. Each word that has been used in this dissertation is now represented by a vector of 17 400 features. In other words, there are 17 400 different words that occur at least $4 \cdot 10^3$ times and at most $1 \cdot 10^6$ times in the corpus. We have been running some experiments between different frequency ranges such as in between the range of 1000 and 1 000 000. Finally, we have reached to the range in between $4 \cdot 10^3$ and $1 \cdot 10^6$ as the best range to construct the vector representation .

Finally, positive point-wise mutual information (PPMI) has been used as a degree of co-occurrence or normalization method to weight the vector values. PPMI is known by giving better results than raw co-occurrence probabilities in different studies (e.g. (Bullinaria and Levy, 2007; Bullinaria and Levy, 2012), as it has been explained in the previous section). The PPMI formula is shown in Equation 2.11. In the equation, a context word is defined as c and a target word as t.

This vector representation framework has been followed for both German and English words in our studies. This framework has taken from Bullinaria and Levy (2007) and Bullinaria and Levy (2012) as it has explained in the previous section.

2.5.2 Word Embedding

Word embedding is used for word representation like distributional features. However, word embedding has dense word representation features regarding distributional features representation. Word embedding can be called the latest version of distributional word representation. Simply, word embedding is a words-dense vector representation.

The neural network is one of the methods that the word embedding uses to compute words representation. The neural network is a computing system that mimics the way in which the human brain operates. It works based on layers of interconnected nodes which are called neurons. In a neural network, a *neuron* is a simple mathematical function capturing and transmitting a information from one neuron to another. The neural network interconnected group of nodes have shown in Figure 2.1. In the figure, the arrow represents an information flow from the output of one neuron to the input of another.

Word2Vec is an example of a well-known algorithm which uses the neural network architecture for word embedding. Word2vec has a number of models that are used in order to produce word embedding. These models follow two-layer neural networks that are trained to reconstruct the context or features of a word (Mikolov et al., 2013b). Two of the most successful and acknowledged recent models are the Skipgram and continuous bag-of-words (CBOW) models which are included in the word2vec library. Skipgram uses the current word *w(t)* to predict the surrounding

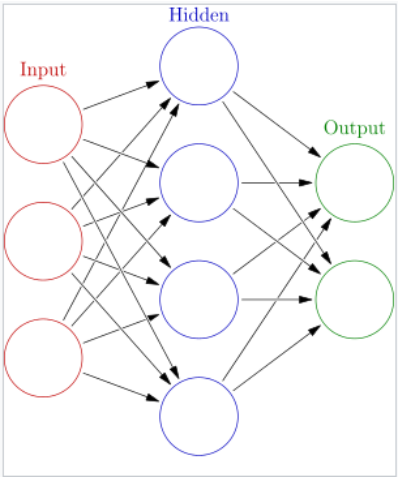

FIGURE 2.1: Neural network - interconnected group of nodes. *Source: Wikipedia*

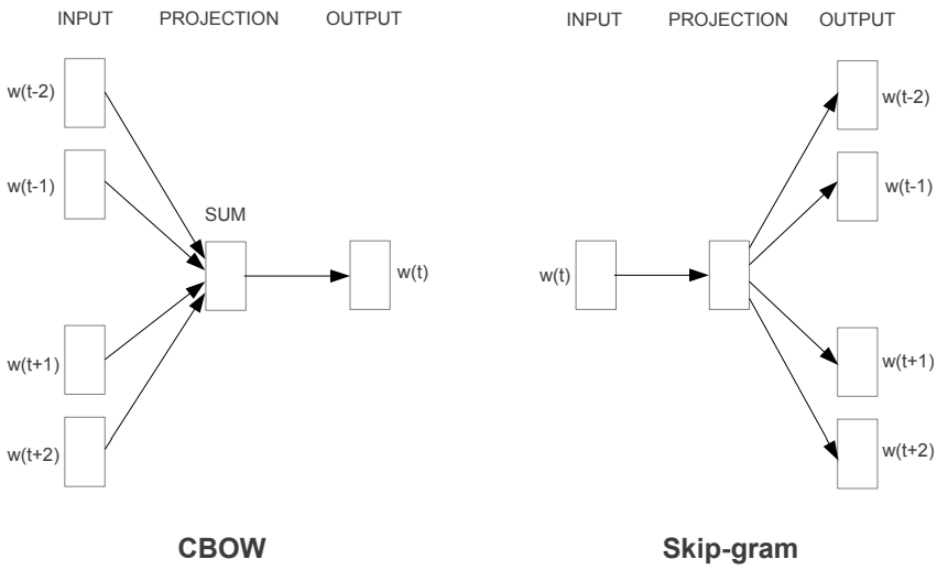

FIGURE 2.2: Source: (Mikolov et al., 2013b).

window of context words *w(t-2), w(t-1), w(t+1),w(t+2)*. CBOW predicts the target word *w(t)* from a window of surrounding context words *w(t-2), w(t-1), w(t+1),w(t+2)*. The Skipgram and CBOW approaches can be seen clearly in Figure 2.2. These models can be called shallow neural networks. Shallow neural networks are neural networks that usually have only one hidden layer as opposed to deep neural network which has multiple hidden layers.

The other well-known word-embedding algorithm is GloVe, which works slightly differently. GloVe is an unsupervised learning algorithm. The GloVe model training is performed on aggregated global word-word co-occurrence from a corpus (Pennington, Socher, and Manning, 2014). In general, there is no need for deep neural networks in order to build good word embedding.

The two common differences of word embedding and distributional representations are the context vector size and the computation that they follow to construct the vector for the word representation. Word embedding has shorter context features regarding distributional features representation. While the word embedding

vector size is in between the range of 200-500, the distributional representation vector size is in between the range 5000 - 30000 features on average. For this reason, the computation time of word embedding is faster than the distributional vector representation.

There is no qualitative difference between the current predictive neural network models and count-based distributional semantics models which is the distributional representation. Rather, they are different computational means to arrive at the same type of semantic model.

Chapter 3

Semantic Similarity

3.1 Introduction

For a large number of applications, where distributional semantic models were successfully applied on (such as question answering, document clustering, paraphrasing), finding semantic similarity between words is the base task. Semantic similarity is the metric of distributional similarity. The main idea of distributional similarity is that words that appear in similar contexts are likely to have a similar meaning, as it has explained in the previous chapter. In distributional semantics, words are represented by a vector which has been constructed by aggregated context features. Then, the similarity of words can be computed by comparing their context features. Thus, it is possible to predict whether two words are synonymous or similar with respect to their semantic relations.

This chapter explains our supervised learning approach study on semantic similarity. Semantic similarity is one of the central tasks in computational lexical semantics. The task is to make a decision on whether two words are semantically related or not. A simple approach is to compute a similarity distance between the words using their vector representation and learn a threshold. The threshold considers that the words are semantically related when their similarity distance is above the threshold, otherwise, unrelated. The similarity distance measure can be the unsupervised methods such as the Cosine or the Euclidean distance. In our study, a supervised learning approach has been proposed for the task. The approach is to train a classifier on the pair of words examples using supervised learning algorithms. To do so, each pair of words has to be represented by a distributional vector. The pairs vector representation can be constructed by methods such as mathematical operations. Then, the supervised learning algorithms use the pair of words vector representation as an input, directly, to train the classifier. Finding the right method also plays a big role in representing the pairs with a good distributional vector and training a well-performing classifier.

Different methods have been studied to construct a vector representation for the pair of words. Some of the main methods are the simple mathematical operations such as addition and multiplication. These methods have been used to construct a vector representation for the pair of words. These methods have found better comparing the approach which combines different types of features to represent pairs. The approach, which constructs the pair of words vector representation by combining different types of functions, proposed by Turney (2013b). This approach has explained in detail in the following sections. All these pair of words vector representation methods performance has been evaluated by the performance of the trained classifiers. The classifiers have been trained by using the pair of words vectors as an input. Then, the classifier which has trained by the pair of words vectors which

have constructed by the pairwise multiplication and addition outperformed the Turney (2013b) approach and also the unsupervised similarity distance measures.

In distributional similarity, studies have been done on very small datasets; For example, the TOEFL data, presumably the most often used dataset in distributional similarity. This dataset has only 80 questions. Therefore, constructing large datasets which simulate the real problem have been part of this study. Thus, using large thesauri like Eurovoc Office for Official Publications of the European Communities (1995) is one of our choices to construct the large datasets. Eurovoc is a multilingual thesaurus maintained by the Publications Office of the European Union. It is available in 24 official languages of the European Union such as German, Swedish, Romanian, Polish. It focuses on the law and legislation of the European Union[1]. It is developed by the cooperation of the European Parliament, the European Union Committee and the Publication Office. The other thesaurus that we chose for our study, is Standard-Thesaurus Wirtschaft (STW) thesaurus[2]. It contains vocabulary for all economic topics (ZBW - Leibniz Information Centre for Economics, 2014). In addition, it includes technical words from law, sociology or politics areas. It contains keywords under 6,000 and more than 20,000 additional synonyms. Constructing datasets from thesauri has two advantages: 1. It is easy to sample large amounts of related (and unrelated) words, and 2. They are constructed independently of any specific semantic task.

This study has been done on six different datasets which have constructed from the above thesauri (Eurovoc and STW) and another source (Bullinaria and Levy, 2007). These datasets contain English and German pairs of semantically related and unrelated pair of words. The four datasets contain the English pair of words. From these four datasets, three of them have constructed from a Eurovoc thesaurus. The other one has been constructed from Bullinaria and Levy (2007) dataset. The dataset from Bullinaria and Levy (2007) is called semantic category 53 (SC53). It contains 530 semantically categorized words under 53 categories. The remaining two datasets contain German pairs of related and unrelated words which have constructed from the STW thesaurus. These six datasets' construction have been explained in Data description Section 3.3.1 in detail.

Since thesauri are also used for automatic indexing and for full-text retrieval, it is important to know all possible terms that refer to a certain concept. Therefore, the extension of thesauri with more labels for each concept is an important task in the maintenance of these vocabularies. The semantic relation, that has been considered in our study, is relatedness of terms in thesauri for intellectual document classification. Thus our findings can directly be applied for the maintenance and extension of such thesauri.

The rest of the chapter is organized as follows. Section 3.2 explains the pairwise feature construction. We discuss the experiment data, supervised method and evaluation in Section 3.3. In Section 3.4 and 3.5, we discuss the result and conclusion respectively.

3.2 Methods

The task that has been considered, is deciding whether a pair of words are semantically related or not. In this study, each word of the pair has been represented by context vectors first. The context vector construction of each word has been carried

[1] http://www.psp.cz/en/sqw/hp.sqw?k=2035

[2] http://zbw.eu/stw/version/9.02/about.de.html

out by following the procedure or framework which has been explained in Chapter 2 Section 2.5 in detail. Then each word has been represented by 17 400 context vector size.

One of the well-known approaches that can be used to identify the semantic relatedness of pairs is computing the distance between the two words' context vectors by using distance measures such as the cosine. For the cosine similarity method, training a classifier is nothing more than finding an optimal value with which to split the pairs which are related and not related. This approach learns the optimal value (or can be called a *splitter*) to classify the pairs. Then, we can have a system which computes the semantic similarity distance between the context vectors of the pair and classifies if the distance is above the splitter or not. If the pairs distance is above or equal to the splitter, they are semantically related; otherwise, they are not related.

In our study, we have been proposed to represent each pair by a vector, and train a classifier on the vector. Two different types of approaches have been investigated to represent the pair of words by a vector. These vector representation approaches are: 1. simple mathematical operations such as addition and multiplication which can be applied to the context vectors of the words of the pair, 2. *SuperSim* method from Turney (2013b) which aggregate different types of features using different functions. Both approaches have been explained in detail in the following sub Section 3.2.1. After representing each pair by two different types of vector representations using the two approaches, two types of classifiers have been trained on the supervised learning algorithm which is the support vector machine (SVM) algorithm using the pairs' vector representation as an input. The two types of classifiers are with respect to the type of the vectors that the pairs have constructed. The classifiers decide whether the pair of words are semantically related or not.

The two types of classifiers (from the mathematical operations and from aggregate features vector representation) performance has been compared between each other and with cosine method classifier as well. Cosine has been used as a baseline to evaluate the performance of our supervised method.

3.2.1 Pair of Words Vector Representation

One of the first proposed approaches that represents the pair of words by distributional vector is using simple mathematical operations. For this approach, four mathematical operations have been proposed. These are *Addition*, *Subtraction*, *Multiplication*, and *Binary*. These operations have explained below with their equations. The equations include the following annotations: $\overrightarrow{A}$ and $\overrightarrow{B}$ represent each word of the pair context vectors; $\overrightarrow{A} = \langle a_1, a_2, a_3, ..., a_n \rangle$ and $\overrightarrow{B} = \langle b_1, b_2, b_3, ..., b_n \rangle$. $\overrightarrow{V}$ represents the pairs vector representation.

- **Addition**: compute the sum of the two words context vector element wise and consider the sum as a vector representation of the pair.

$$\begin{aligned} \overrightarrow{V} &= \overrightarrow{A} + \overrightarrow{B} \\ &= (a_1 + b_1, a_2 + b_2, a_3 + b_3, ..., a_n + b_n) \end{aligned} \tag{3.1}$$

- **Subtraction** : compute the difference between the two words context vector element wise and consider the difference as a vector representation of the pair. The subtraction equation is an absolute value to remove the negative sign because the negative sign may lead the representation in a different unknown

direction.

$$\begin{aligned}\overrightarrow{V} &= \overrightarrow{A} - \overrightarrow{B} \\ &= (|a_1 - b_1|, |a_2 - b_2|, |a_3 - b_3|, ..., |a_n - b_n|)\end{aligned} \tag{3.2}$$

- **Multiplication**(Point-wise or Hadamard product): closely related to cosine if the length of the vectors is normalized. Cosine is the sum of the elements of their Hadamard product.

$$\begin{aligned}\overrightarrow{V} &= \overrightarrow{A} \odot \overrightarrow{B} \\ &= (a_1 \cdot b_1, a_2 \cdot b_2, a_3 \cdot b_3, ..., a_n \cdot b_n)\end{aligned} \tag{3.3}$$

- **Binary**: The binary vector $\overrightarrow{V}$ of two vectors $\overrightarrow{A}$ and $\overrightarrow{B}$ is defined by setting $v_i = 1$ if $a_i > 0$ and $b_i > 0$, otherwise $v_i = 0$.

$$v_i = \begin{cases} 1 & \text{if } a_i > 0 \text{ and } b_i > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

These mathematical equations have been applied to the context vector of the words of the pair. These vectors have been normalized or weighted by positive point-wise mutual information (PPMI) method, as it has explained in Chapter 2 Section 2.5. For further investigation, the equations have been applied on the vectors which are the unit vector of the words context vectors. A unit vector is any vector with a magnitude length of one. The vector can point in any direction, as shown in the following Figure 3.1. To find a unit vector of a vector, it is simply dividing the

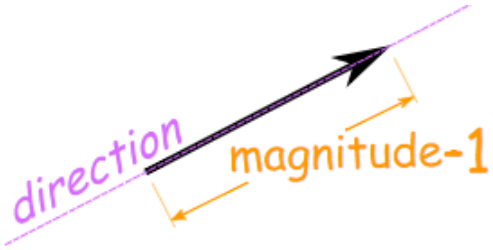

FIGURE 3.1: Unit Vector

vector by the magnitude of the vector as shown in the following Equation 3.5.

$$\hat{u} = \frac{\overrightarrow{v}}{|v|} \tag{3.5}$$

In mathematics, magnitude represents the size of a mathematical object. The magnitude equation has shown as follow in Equation 3.6; Where a_1, a_2 ... a_n are the n number of components of a vector.

$$|v| = \sqrt{\sum_{i=0}^{n} {a_i}^2} \tag{3.6}$$

In our study, the other type of vector that has been used on the mathematical equations as an input to represent the pairs, besides the context vector, is unit vector. Unit vector of each context vector has been computed before applying the mathematical equations to construct the pair of words vector representation. The unit

vector has been considered on the first three equations (Addition, Subtraction and Multiplication); Because the fourth equation (Binary) output vector is boolean. Usually, we use a unit vector to compute the angle between vectors[3]. In general, seven different types of a pair of words vector representations which have constructed by the simple mathematical operations have been considered to train a classifier on SVM by taking the vector representations as an input directly. In other words, seven different classifiers have been trained by giving the seven different vector representations as an input to SVM. This explanation has cleared in Table 3.1.

Types of vectors	Mathematical Operation			
	Addition	Subtraction	Multiplication	Binary
Context vector	✓	✓	✓	✓
Context vector + Unit vector	✓	✓	✓	-

TABLE 3.1: The seven different pair of words vector representations that the SVM has been using as an input to train the seven different classifiers.

The second approach that has been considered to construct a pair of words vector representation is the Turney (2013a) approach. This approach is called *SuperSim*. The approach constructs vector representation for a pair of words which identifies related pairs. The approach represents a pair of words by aggregating four different types of features, which are all based on frequencies in a large corpus. The type of features is logarithm frequency *LF*, which computes each word of the pair logarithm frequency, *PPMI* between the two words of the pair, the similarities of the two words in domain space *Dom*, and the similarity of the two words in function space *Fun*. In Turney (2013a) study, *SuperSim* studied on n size of the tuple (such as word pairs, phrases, or sentences), and the four different types of features vector represent the tuples. In our study, the approach has been used only for the word pairs. Therefore, the approach explanation focuses only on the word pairs tuple.

To explain the four types of features, let us use the variables which have used to explain the mathematical operations above. In addition, we have defined more variables. w, A, and B represent a word and each word of the pair respectively. The first type of features is represented as LF. LF features are the logarithm log frequency $freq$ of each word of the pairs. Let $freq(w)$ be the frequency in a corpus. If the word is not available in the corpus, then the frequency of the word will become zero. Therefore, the equation adds one to the frequency of the word before the log computation. Because, $log(0)$ is undefined. The equation is shown as follow in Equation 3.7:

$$LF(w) = log(freq(w) + 1) \tag{3.7}$$

Equation 3.7 has been applied on each word of the pair $LF(A)$ and $LF(B)$. Then, the pairs have represented by the two features $LF(A)$ and $LF(B)$ from LF.

The second set of features are $PPMI$. The $PPMI$ features are the PPMI of each word of the pair. For this set, simply, the $PPMI$ matrix that we have constructed for this project and others have been used. The PPMI matrix construction has explained in detail in Chapter 2 Section 2.5. This type takes the $PPMI$ of each word of the pair by considering one of the words as a target word (A *or* B) and the other word as a context word (A *or* B) which occurred on the right or left side of the target word

[3] https://www.vcalc.com/wiki/vCalc/V3+-+Angle+between+vectors

in the corpus. To find the PPMI of each word in the matrix, the target word corresponds to the row, and the context word corresponds to the column. The original *SuperSim* method considers the position (right and left) of the context word with the target word in a corpus to compute the PPMI features type. However, in our case, the context word position of each word is not separated by left and right. The words ppmi have been constructed by considering the left/right contexts, but, without considering the position. For example, *x, y, z, b* and *b, z, y, h* are two phrases in a corpus. Let us consider (y, z) as a pair. The original *SuperSim* method counts the co-occurrence of target word y with context word z on the left as one PPMI and the target word y with the context word z on the right side as another PPMI. The same procedure applies again by considering z as a target word, and y as a context word. At the end, the pair is represented by four PPMI features set. In our case, both the right and left co-occurrence of y with z are summed up together. Then, we have applied the same procedure again by considering the context word as a target word and the target word as a context word. Finally, we have two PPMI values for each pair on this features set. The mathematical operation has been shown as follows:

$$\begin{aligned} &PPMI(A, B, left/right); \text{ where } A \text{ is target word and } B \text{ is context word} \\ &PPMI(B, A, left/right); \text{ where } B \text{ is target word and } A \text{ is context word} \end{aligned} \tag{3.8}$$

The third set of features are referred to as *Dom*. *Dom* is the domain space which finds the similarities of two words by following the idea that the domain or topic of a word is characterized by the nouns that occur near to it. For *Dom* type, one matrix has been construed which has only noun contexts as a feature for each word instance from the previous *PPMI* matrix. The previous *PPMI* matrix is the matrix that has been used for the whole experiments, and which has been constructed to represent each word by the framework that has explained in Chapter 2 Section 2.5. The noun features have been extracted from the contexts which have frequency between 4000 and 1M in the corpus. Since the previous *PPMI* matrix already have the words with the noun contexts, we have collected the word-noun context from this matrix. SVD has been applied on the original *PPMI* matrix during the construction of the matrix; But, SVD did not help to improve the vector representation when the performance has been evaluated on TOFEL and semantic classification tasks. Therefore, SVD has not been applied to the newly-generated word-noun context matrix as well. The similarity of words in domain space is Dom(A, B , k, p) that is computed by extracting the row vectors in word-noun context matrix that correspond to the words A and B , and then calculating their cosine $cos(A, B)$. The initial value of k = 100 and p = 0.0 ; where, k is the number of latent factors in the word-noun context vector representation of words (or latent feature size), and p raises the vector values in $\sum_k$ to the power p. Then, the following loop has been applied to calculate each k and p parameters value and compute the cosine of the pair:
for k=100:1000:+100 and p=0:1:+0.1 (together 110 features).

The fourth set is the function space type features *Fun*. *Fun* have also been aggregated with the other three types of features to add the function or role of a word from the syntactic context that relates the word to the verbs that occur near to it (Turney, 2013a). This type is exactly the same as the third set. However, unlike Domain space, function space type considers verbs only instead of nouns to construct the PPMI matrix. The word-verb context matrix has been constructed just like word-noun context matrix, but verb has been used instead of noun in the function space. Then, the Domain space type formulas have been applied for the function space type Fun(A, B

, k, p) as well. However, the formulas have applied to word-verb context matrix (ppmi matrix). Finally, the four sets of features have been aggregated together as one vector to represent the pair of words. The total size of the vector that represents the pair is 224. Each type of feature's size is shown in Table 3.2.

Feature set	Size of set
LF (w)	2
PPMI(A, B, handedness)	2
Dom(A,B,k,p)	110
Fun(A,B,k,p)	110

TABLE 3.2: Each type of features size of *SuperSim* approach

In general, the above two approaches (mathematical operations and *SuperSim*) have been used to represent pairs by a vector. Then, SVM has been using the vectors as an input to train the pair of words' classifier. The accuracy of the two pairs vector representation approaches have been investigated by the performance of the classifier which has been trained on the pair vectors. The accuracy results have been compared between each other. In addition, the performance of the classifier has been compared with the baseline cosine method as well in order to find the best vector representation and classifier which identify the semantically-related pair of words. The complete results are explained in the result Section 3.4.

3.3 Experiment

In this section, the datasets and the experimental setup are described.

3.3.1 Data description

In this study, all words of the training and test set have been represented by context vectors. The context vector of each word has been constructed using two different corpora: 1. ukWaC for English words, and 2. deWaC for German words. The corpora and also the vector representation have been explained in detail in Chapter 2 Section 2.5.1. Therefore, this section focuses only on the explanation of the experiment datasets construction. The six training and test datasets are constructed from two large thesauri and from a dataset which has introduced in Bullinaria and Levy (2007) study.

First, we have constructed three datasets from a Eurovoc thesaurus and two datasets from STW thesaurus by considering their tree structure. Then, by using these datasets, we have constructed the experiment datasets which contain a pair of words. In total, we have constructed five datasets which contain a pair of words from thesauri. The sixth pair of words dataset has constructed from the Bullinaria and Levy (2007) dataset (*SC*53). These pairs of words (from the six datasets) have co-hyponym semantic relation, because they have constructed from the datasets which contain words with hypernym relation information. The first two columns of Table 3.3 can be an example of the datasets which contain the words with hypernym semantic relation. Column one contains a category called *color*. *Color* is a hypernym of *blue, black, yellow.....* Or we can say *blue, black, yellow....* is a hyponym of *color*, as the definition of semantic relations explained in Chapter 1. The third column of Table 3.3 can be an example of the pair of words datasets which have a co-hyponym semantic relation. For example, *mile - centimeter, inch - mile, red - pink...* have been

constructed from column one and two. The word *mile is a co-hyponym of centimeter, inch is a co-hyponym of mile, and red is a co-hyponym of pink*. The last column of Table 3.3 shows example of unrelated pair of words. These datasets' construction has been explained below.

The Bullinaria and Levy (2007) dataset (*SC*53) is prepared for a semantic classification task. It contains 530 words which are taken from 53 semantic categories (10 words for each category). From the SC53 dataset, 3504 semantically-related and unrelated pairs of words have been constructed. 1752 pairs of words belong to the same category, while 1752 randomly chosen pairs of words belong to two different categories. The pair of words which belong to the same category have been considered as positive examples or semantically-related pairs of words since they share the same semantic category. The pairs which have reconstructed from different semantic categories, have been considered as negative examples. This is because the words of the pair have different semantic categories. Table 3.3 shows examples of the pairs, how they have constructed from the SC53 dataset. The constructed pair of words dataset is also referred to as SC53, since they are constructed from the SC53 dataset.

Semantic Categories (SC53)		**Examples**	
Colors	**Measures**	**Related Pairs**	**Unrelated pairs**
blue	mile	mile - centimeter	meter - green
black	foot	inch - mile	blue - acre
yellow	inch	meter- centimeter	blue - mile
red	yard	millimeter - acre	inch - yellow
green	meter	furlong - acre	millimeter - brown
brown	centimeter	red - pink	brown - furlong
white	millimeter	blue - green	purple - centimeter
pink	acre	blue - purple	red - kilometer
purple	furlong	white - yellow	inch - pink
orange	kilometer	black - brown	meter - red

TABLE 3.3: SC53 pair of words construction example

The remaining five training and test datasets are constructed from thesauri Eurovoc and STW. These thesauri are explained in the introduction section of this chapter. Since thesauri are organized hierarchically, similarity at different levels can be defined. Both very fine-grained level and broad level similarity can be used, and all terms belonging to the same branch of the hierarchy can be considered as similar. However, since each thesaurus has some focus domain that is worked out very detailed and other areas that are modeled much more coarse-grained, in the core of the thesaurus two terms denoting the same concept will be real synonyms, while in other areas quite different words can refer to the same concept. Keeping this in mind, it is understandable that an automatic approach will never be able to decide whether two terms denote the same concept without errors.

Three datasets have compiled from the Eurovoc Thesaurus. Eurovoc is a multilingual thesaurus developed by the European Commissions Publications Office as a controlled vocabulary for the intellectual indexation of documents Office for Official Publications of the European Communities (1995). The Eurovoc thesaurus is divided into 127 micro-thesauri. From each of these micro-thesauri, 528 in total, top-level concepts have been taken as semantic categories. For each category, all narrower concepts have been collected and their preferred and alternative labels have been considered as terms for that category. Then all terms that belong to more than one

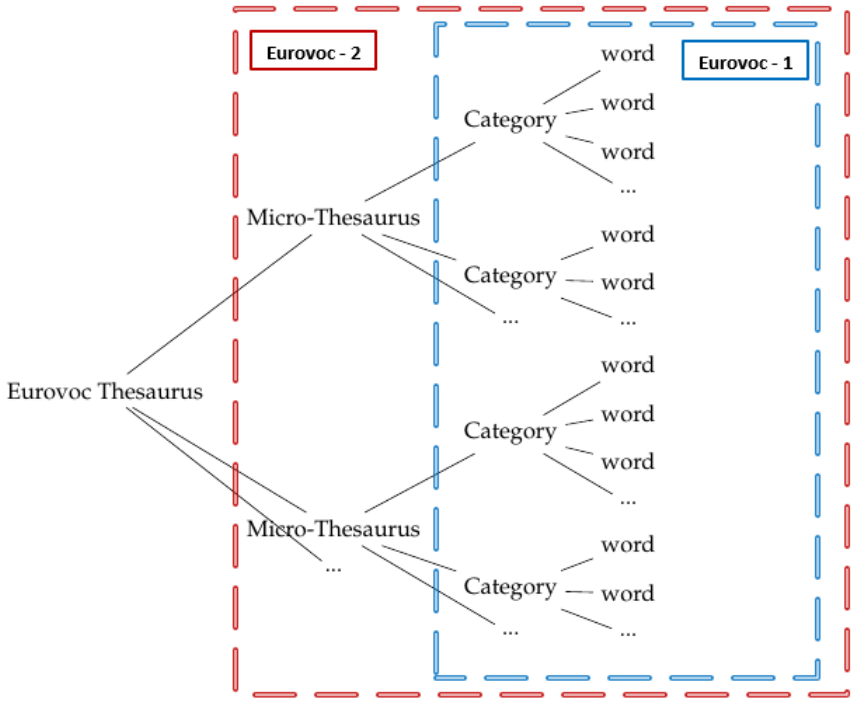

FIGURE 3.2: Eurovoc Thesaurus datasets construction tree

category or that consist of more than two words have been removed. Finally, all categories which contain less than 10 terms, have been removed. Now, 190 categories with a total of 2386 terms are left. The largest category consists of 44 terms. From this dataset two set of pairs have been constructed. The first set has 10 000 pairs of words belonging to the same category, and the second set has 10 000 randomly chosen pairs from two different categories. We refer this set of pairs as *Eurovoc-1*, since the terms are equivalent by going up one level in the Eurovoc concept hierarchy. Furthermore a collection of pairs have been built by selecting 10 000 pairs of words from the same data set where both words are taken from the same micro-thesaurus and 10 000 pairs taken from two different micro-thesauri. This set is referred as Eurovoc-2. Figure 3.2 shows the two datasets pair of words construction hierarchy.

Finally, pairs of words from the original Eurovoc thesaurus (before applying the modification for Eurovoc-1 and Eurovoc-2) have been sampled by taking preferred and alternative labels for the same concept as synonymous terms, and pairs that are used as labels for different concepts as non-synonym pairs. For the negative examples, an equal distribution of easy and difficult pairs have been considered. Thus, 20% pairs which are words from concepts with a distance of 1 step have taken by using any specified thesaurus relation. Further 20% has been taken from concepts with a distance of 2 steps, and so on. For the last 20%, pairs of concepts with a distance of at least 5 steps have been used. In all cases it has been ensured that no shorter path exists. For this set, 2175 synonymous and 2335 non-synonymous words have been taken. This set of pairs are referred as *Eurovoc-0*. The hierarchical tree of this dataset is same with the dataset *STW-0* which has constructed from STW thesaurus. Therefore, we can simply refer *STW-0* in Figure 3.3 to show the hierarchical tree structure of Eurovoc-0 dataset. *STW-0* is explained in the following paragraphs. In the figure, *descriptor* means *concept* in Eurovoc-0 case.

The three datasets (Eurovoc-0, Eurovoc-1 and Eurovoc-2) are consisting of only two single words pair. Moreover, a word which occurred at least once in the ukWaC

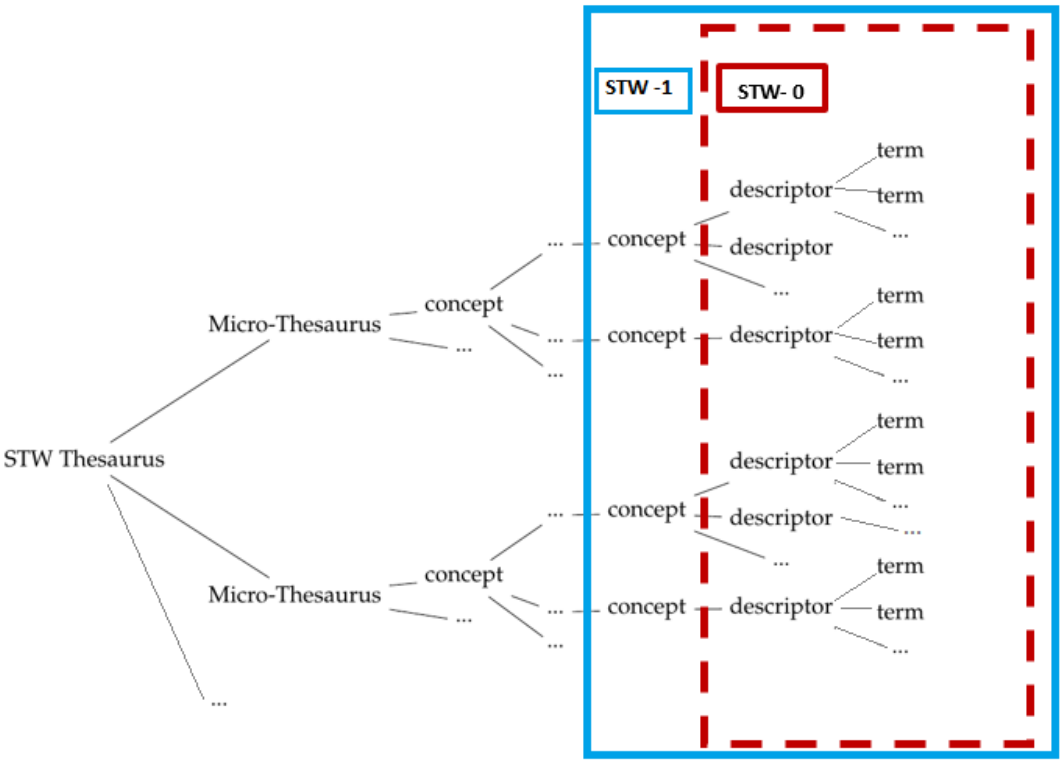

FIGURE 3.3: STW Thesaurus datasets construction tree

corpus has been selected.

The German thesaurus on business and economics *Standard-Thesaurus Wirtschaft* (STW) are used to derive the German word pairs. The STW is divided into 7 sub-thesauri (Economics, Business Administration, Economic Sectors, Products, Neighboring Sciences, Geographical Terms, General Words). Each part consists of a hierarchy of notations and descriptors. Descriptors have broader, narrower, and more related terms. We have been taking all terms (i.e. labels from descriptors) from the 6 sub-thesauri (leaving out the sub-thesaurus with general terms) that belong to only one notation and consist of at most 2 words. Subsequently, all words belonging to a notation with less than 5 terms in the sample have been removed. This gives us 419 concepts (one concept for a notation) with a total of 11 599 terms. There are 5 concepts with over 100 terms. The largest concept has 233 terms. From this set, a set of a random pair of words have been selected, 10 000 from the same concept and 10 000 from different concepts. We restricted the selection of words to words occurring at least once in deWaC. This set is referred to as STW-1.

Finally, pairs of words where the terms are labeled for the same or different descriptors have been selected. This dataset has 10 000 positive and 10 000 negative pairs. This set is referred to as *STW-0*. The distribution of negative pairs of this dataset has been collected in the same way that the Eurovoc-0 dataset has been controlled. The hierarchical tree structure of these two datasets is shown in Figure 3.3.

The pair of words datasets size is shown in Table 3.4 with their positive and negative classes.

3.3.2 Supervised Similarity Learning

The supervised learning method has been used to train the classifiers by taking the pairs distributional features representation as an input, and classify the semantically

TABLE 3.4: Pair of words datasets size

Datasets / Classes	SC53	Eurovoc-2	Eurovoc-1	Eurovoc-0	STW-1	STW-0
Positive	1 752	10 000	10 000	2 175	10 000	10 000
Negative	1 752	10 000	10 000	2 335	10 000	10 000

related and unrelated pair of words. As a supervised learning method, Linear SVM has been used from the liblinear package. Linear SVM takes the pair of words vector representation as an input directly and train a classifier (build a model). Liblinear is very efficient and fast for training large-scale problems (Fan et al., 2008). In our study, the number of features is 17,400. Therefore, it is reasonable to use liblinear. When we use liblinear, the main parameter that we need to tune is parameter C. Before building the model, the hyper-parameters of the model have been tuned using grid search python script (grid.py) from LIBSVM.

The grid search script has been used to find the best C parameter value. A number between the range 0 and 20 in step 0.05 have been tuned to find the best C parameter value. Then, the C parameter value which has been found as the best value, has been used in Linear SVM in order to build the model. For example, the following command has used to tune the C parameter:
./grid.py -v 10 -log2c 0,20,0.05 -log2g null -svmtrain ./train Eurovoc.train.
Since liblinear does not tune gamma, it becomes *null* as shown in the command *-log2g null*. The grid search script uses cross validation technique to estimate the accuracy of each parameter combination in the specified range and helps to decide the best parameters. Therefore, the C parameter value has been tuned by applying cross-validation to the training set with argument $-v$. Then, the best C parameter has been applied to the training set to build the model.

Finally, the model has been evaluated on the test set. For evaluation, ten-fold cross-validation has been utilized. The entire dataset has been split into 90% training and 10% test sets. With the given split percentage, ten training and ten test sets have been prepared. Then the model has been trained and evaluated on each ten training and test set. This is explained in detail in the following section.

3.3.3 Experiment Setup

For all experiments, ten-fold cross-validation has been used on the datasets in order to build the classifier models. Cross-validation is a technique that we use to evaluate predictive models by partitioning the dataset into a training set to train the models, and a test set to evaluate them. By using a cross-validation technique, the dataset can be partitioned into k folds (parts). In our study, the 10-fold partition has been applied to the datasets. Our dataset's class examples size is not the same. Therefore, stratified sampling methods have been followed to partition the experimental datasets. 10% were considered for the test while 90% was considered for the training set. Thus, ten training and ten test datasets have been generated. As an example of stratified sampling, let us take a dataset which has three classes and each class has different sample/examples size (150, 200 and 250). If we want to partition this dataset to 50% training and 50% test set, we take the following sample from each class: 150*0.5= 75, 200*0.5=100 and 250*0.5= 125. In our study, we have followed the same procedure when splitting the experimental datasets.

3.4 Results

This section shows our study experiments final result. Table 3.5 shows the experiment's result. The overall idea of the study is to investigate the supervised and unsupervised learning approaches performance on classifying semantically related pair of words task. This idea has studied by Weeds et al. (2014), as well, on hyponym and the co-hyponym semantic relation of a pair of word prediction tasks. However, their approach has shown good performance on the hyponym semantic relation prediction task as explained in the related works of distributional similarity in Chapter 2 Section 2.2. Our study, specifically, focuses on the co-hyponym semantic relation of pairs task. For this study, six datasets have been considered; four English and two German pairs of words datasets. In total 48 classification experiments have been done on both English and German pairs, and four more classification experiments on only English pairs which are the experiments on *SuperSim* vector representation approach. The average accuracy results from ten-fold cross-validation are given in Table 3.5. The results confirm the outcome of the experiment from Hagiwara (2008). Hagiwara (2008) proposed an approach to automatic synonym identification based on supervised machine learning and distributional features. In our study, we have also shown that using distribution features directly in the supervised learning method such as SVM can train a good classifier which classifies semantically-related pair of words. However, Hagiwara (2008) distribution feature construction for the pair of words is totally different from our vector representations. For the pair of words, Hagiwara (2008) defined distributional features using the co-occurrence as explained in detail in Chapter 2 Section 2.2. The feature value is determined to represent the degree of commonality of the context shared by the word pair.

As unsupervised learning approach, cosine distance measure has been considered and a *splitter or threshold* has been trained to classify the pairs on the six datasets. As the result table shows, the cosine method results are the lowest results comparing many of the rest supervised classifiers. The results, which have been obtained by cosine similarity method, have been considered as a baseline. Because cosine is usually considered as the best similarity measure for classification of context vectors (Bullinaria and Levy, 2007; Bullinaria and Levy, 2012; Kiela and Clark, 2014). In addition, we have also investigated other similarity distance measures such as Euclidean and Jaccard with the cosine method on the SC53 dataset. Then, the cosine has been found as the best similarity distance measure from the rest unsupervised methods.

From the total 52 classification experiments, the 46 classifiers which have been trained on SVM using a pair of words vector representation yields better results comparing cosine distance similarity measure. The pair of words representation that the 46 classifiers have been using as an input, is constructed by the two approaches (mathematical operations and *SuperSim*). From these two types of vector representation, the pairs representation which have been constructed using the simple mathematical operations have been performed better than the *SuperSim* approach when we evaluate the classifiers which have been trained using them. *SuperSim* approach is another baseline that gave good results for a pair of words and phrase similarity in the (Turney, 2013b) study. When we compare the *SuperSim* performance with cosine, it has the same performance on the SC53 dataset. However, it performs better than cosine for the pairs which have broad similarity relation like the Eurovoc-2 dataset pairs, as shown in the result table. But, cosine performs better than SuperSim for the pairs with the tight similarity relation like Eurovoc-1 and Eurovoc-0.

From the four simple mathematical operations, in general, the experiments have

shown that addition and multiplication are the best choices to construct vector for the pair of words representation, and use the representation to train a classifier using SVM. As the result table shows, the classifiers which have been trained on the vector that has been constructed using addition and multiplication which has been outperformed the other classifiers. These operations compute better vector representation when the feature vectors are normalized (by unit vector) before the operations computation on the context vectors of the words of the pair. As the result table shows that some classifiers perform better on one dataset, but not on the others. Therefore, the classifiers' average performance has been calculated on the six datasets so as to find the best overall classifier. The average computation of all classifiers has shown in the last column of the result table. When we see the overall average results, the classifier which has trained on SVM by taking the pair of words vector representation as an input (which has been constructed by multiplication operation with the unit vector normalization) has outperformed all other classifiers to classify the semantically-related pairs well.

Method		**Datasets**						
	Methods	**SC53**	**Eurovoc-2**	**Eurovoc-1**	**Eurovoc-0**	**STW-1**	**STW-0**	**Average**
Split	**Cosine**	0, 87	0,71	0,77	0,75	0,65	0,65	0,73
Linear SVM	**SuperSim**	0,87	0,77	0, 69	0,73			0,77
	Binary	0,96	0,99	0,85	0,71	0,80	0,63	0,82
	Addition	0,94	**1,00**	0,92	0,65	0,81	0,53	0,81
	Subtraction	0,99	**1,00**	0,94	0,73	0,81	0,55	0,84
	Multiplication	0,96	0,99	0,86	0,72	**0,82**	0,66	0,84
	Addition (unit vectors)	**0,99**	0,99	**0,95**	0,74	0,81	0,55	0,84
	Subtraction (unit vectors)	0,99	**1,00**	0,94	0,73	0,81	0,55	0,84
	Multiplication (unit vectors)	0,97	0,92	0,88	**0,84**	0,80	**0,71**	**0,85**

TABLE 3.5: Accuracy of *synonymous* and *non synonymous* word pair classifiers

3.5 Conclusion

The study has suggested learning a SVM model on a pair of words' vector representation to classify a semantically-related pair of words. The pair of words' vector representation needs to be constructed by simple mathematical operations. Learning SVM models on the features which have been constructed by simple mathematical operations, have outperformed comparing similarity distance measures like cosine and also learning a model on the pair of words vector representation which has been constructed by aggregating different types of features. In the experiment, multiplication and addition have been performing as the best methods to construct pairwise features, and learn a model for the pairs relation on SVM. Furthermore, the experiment has shown that the vector should be normalized before the words context vectors are combined by one of the mathematical operations to represent the pair of words.

In general, from the overall results, the classifiers which have trained on the vectors that have constructed by the multiplication method on the normalized vectors,

have outperformed the rest classifiers. Specifically, we can conclude that the classifier with normalized multiplication method can be recommended for the pair of words with tight semantic relation such as the Eurovoc-0 and STW-0 dataset pairs. This research can be applied to the maintenance and extension of thesauri.

Chapter 4

Semantic Classification

4.1 Introduction

This chapter explains the semantic classification of words study in the cases where supervised learning methods are used. Semantic classification of words is, basically, classifying semantically related words into the same category. It is usually based on the semantic similarity of the words using their distributional features information. For example, words which are semantically related can be classified using similarity distance measures such as cosine. Similarity distance measures are commonly used in the nearest neighbor or a nearest centroid (or nearest prototype) classifiers for the semantic classification of words. This approach is an unsupervised learning approach to classify semantically-related words.

In this study, a supervised learning approach is proposed in order to build a classifier which classifies semantically-related words. The classifier is trained on supervised learning algorithms such as Support Vector Machine using the distributional features of the words as an input directly. Moreover, Multi-Relational Matrix Factorization (MRMF) is introduced in distributional semantics to train a classifier for semantic classification of words by following a supervised learning approach. MRMF is a novel approach in distributional semantics.

The study, specifically, is considering a task which classify semantically related words into a large number of semantic categories to investigate and compare the supervised and unsupervised learning approach methods performance. In the study, the classifiers which have been trained on supervised learning algorithms using the distributional features directly as an input have been giving better results comparing the unsupervised learning approaches (i.e. similarity distance measures). The performance of the classifiers which have been trained on SVM and MRMF have related results when they both use only dis- tributional features information as an input. The classifiers from LR algorithm has less performance comparing SVM and MRMF.

In the study, we have considered to integrate the distributional features information with other sources of information for the classifiers better performance. MRMF can easily be extended with more matrices which contain more information from different sources on the same problem. For SVM, we have been using ensemble learning to integrate the different sources of information. Ensemble helps to integrate a set of models which have obtained by learning process on a given problem and build one general predictive model in order to improve predictions. When the SVM use the ensemble method, first, a model is trained on each source of information and integrated by the ensemble method. Then the ensemble method builds one general classification model. When the classifiers train on the integrated multiple sources of information, the performance of MRMF outperforms SVM.

The effectiveness of the novel approach has been demonstrated by using information from WordNet as additional source of information to be integrated with distributional information. WordNet[1] is a large lexical database of English. Thus the study shows, that MRMF provides an interesting approach for building semantic classifiers. The MRMF classifiers (1) gives better results than unsupervised approaches which are based on vector similarity such as nearest neighbor or a nearest centroid, (2) gives related results as other supervised methods such as SVM when only distributional information is used to train the classifier, and (3) can naturally be extended with other sources of information in order to improve the classification results.

This study's findings have shown on two different datasets. These datasets are: 1. A dataset which is used in the literature (Bullinaria and Levy, 2007) to enable compare our results with the results reported in the literature, 2. A larger dataset that is derived from a large thesaurus. The second dataset comes close to practical applications for semantic word classification. Organizing and maintaining thesauri are usually done by trying to keep their thesaurus up to date and frequently adding new terminology to the thesaurus. For each new term, they have to decide at what point it has to be inserted. Automatic semantic classification supports tasks like this. However, the classifier should not only be able to choose from only dozens but rather hundreds or even thousands of semantic classes.

This chapter is organized as follows. Section 4.2 explains the methodology of the work in detail. Section 4.3 explains the multi-relational matrix factorization method in detail. Section 4.4 and Section 4.5 explain briefly the evaluation of the models and explain the results respectively. Finally, Section 4.6 concludes the chapter.

4.2 Methodology

This section explains in further detail the datasets which have been used for the experiments, the feature construction to represent each word of the datasets, and the classification methods.

4.2.1 Data Description

In this study, two datasets have been used as mentioned briefly in the introduction section. The first dataset is the Bullinaria and Levy (2007) dataset which is used in their semantic classification study. This dataset has 53 semantic categories out of the 56 basic semantic categories which are introduced by Battig and Montague (1969). The dataset contains 530 words in total. The words are taken from the 53 semantic categories. Each category has 10 words. This dataset is referred to as *SC*53.

The second dataset is derived from Eurovoc Thesaurus (Office for Official Publications of the European Communities, 1995). It is referred to as *Eurovoc*. This dataset is much bigger than the SC53 dataset. The dataset construction has explained in detail in Chapter 3 Section 3.3.1. This dataset is used to construct the Eurovoc-1 dataset which contains semantically related and unrelated pair of words for our semantic similarity task study in Chapter 3. As a reminder, Eurovoc Thesaurus is a multilingual thesaurus developed by the European Commissions Publications Office as a controlled vocabulary for the manual indexation of documents. The Eurovoc thesaurus is divided into 127 micro-thesauri. From each of these micro-thesauri we took the top-level concepts as semantic categories. In total, they are 528 semantic

[1] https://wordnet.princeton.edu/

categories. For each category, we have collected all narrower concepts and considered their preferred and alternative labels as terms for that category. Then all terms that belong to more than one category have removed. Finally, we have removed all categories which contain less than 10 terms. Now, 190 semantic categories with a total of 2386 terms are left. Further, the dataset is cleaned by removing the words that have very high or low frequencies or those which are not available in the ukWaK corpus. This is because the removed words may not have context words in the corpus during the context vector representation. ukWaK corpus is a corpus which has been used to construct the word representation vectors. Finally, 1447 words with 95 semantic categories are left. Now, each category contains 10 to 44 terms.

These two datasets' sample data are shown in Table 4.1. Table 4.1 shows what the datasets look alike by some examples.

Dataset	Category	Words
SC53	Fruits	Orange, Strawberry, Banana
	Furniture	Chair, Table, Bed
Eurovoc	ACP countries	Bahamas, Barbados,Cameroon
	Health policy	Dispensary, Hospitalization

TABLE 4.1: Some examples of semantic categories with their words from the experiment datasets SC53 and Eurovoc

4.2.2 Representation of words

Two types of representations have been considered to represent each word of the datasets by a vector. The first representation is using the distributional features information which is based on word co-occurrences. This representation have been explained in Chapter 2 Section 2.5.1. Each word has been represented by distributional features as explained in Chapter 2 Section 2.5.1. The second representation is using the lexical information. WordNet[2] has been used to construct this representation for each word. The following subsection explains that how the lexical information has extracted from the WordNet and how the words have represented by the lexical information.

4.2.3 Lexical Information Representation

One of the well known English language databases for lexical information is WordNet. WordNet groups English words into sets of synonyms. The set of synonyms is called *synsets*. The main relation among words in WordNet is synonymy. For example, the relationship between the following words is that of synonyms: *shut* and *close*, *car* and *automobile*. Synonyms are words that have similar meanings or concepts. Synsets are connected with other synsets to form a hierarchy of concepts, ranging from very general to very specific. Some of the relations are:

- hypernyms are the synsets that are more general.
  E.g. canine is a hypernym of dog.
- hyponyms are the synsets that are more specific.
  E.g. dog is a hyponym of canine.

[2] https://wordnet.princeton.edu/

Hyponyms have an "is-a" relationship to their hypernyms. Hypernyms is one the relation that has been considered in our study to represent each word.

In order to classify words into semantic categories, we could directly use the semantic categories of the words from WordNet. However, we do not know the relationship between the WordNet categories and the target categories. The target category is the semantic category that the word has from the Eurovoc thesaurus or from the SC53 dataset. As a result, each word has been represented by the set of all WordNet hypernyms, i.e. the transitive closure of the hypernym relation applied to each possible meaning of the word. E.g. the word *mansion* is represented by the set {artifact.n.01, building.n.01, dwelling.n.01, entity.n.01, house.n.01, housing.n.01, location.n.01, mansion.n.02, object.n.01, physical_entity.n.01, region.n.01, sign_of_the_zodiac.n.01, structure.n.01, whole.n.02b}. The python script of the example is as follows:

```
from nltk.corpus import wordnet as wn
word = wn.synsets('mansion')[1] '''[1] selected to use the 'mansion' \

                    synset which has large number of hypernyms'''
mansion_word = [i.lemmas().name() for i in \
                list(set(word.closure(lambda s:s.hypernyms())))]
```

After collecting hypernyms of each word, all hypernyms of each word have been aggregated and became features n size to represent the words. The n size features are free from duplicated hypernyms from the aggregation. Then, each word has been represented by a vector $\overrightarrow{V}$ which contains boolean values. For example, if hypernyms feature c_i is a hypernym of word w, the value of the vector becomes 1, otherwise 0. The vector construction is shown in the following equation:

$$V_w(i) = \begin{cases} 1 & \text{if hypernyms feature } c_i \text{ is a hypernyme of } w \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

Moreover, the experiment datasets contain words that are not found in WordNet. Therefore, in the SC53 dataset, 520 words have been represented by WordNet vectors out of 530 words. In Eurovoc dataset, 1198 out of 1447 words have been represented by the WordNet vectors. The average number of hypernyms for each word which has been found in the WordNet is 66. The total number of distinct hypernyms that have used to represent words in the SC53 dataset are 2896 and 4938 hypernyms for words in the Eurovoc dataset.

4.2.4 Classification Methods

In this study, three main different methods have been investigated for the semantic classification task. These are the nearest centroid method (unsupervised learning), SVM, and MRMF. LR has also considered as an additional method for further investigation of the well-known supervised learning algorithms.

The nearest centroid is used in Bullinaria and Levy (2012) study as well for the same task. Bullinaria and Levy (2012) used the nearest centroid classifier for classifying words based on distributional features. In this approach, a feature vector (centroid) is constructed for each semantic category by averaging the feature values of all words in the training set which belongs to that category. Then, the cosine distance is computed between the feature vector of the word and each centroid vector

of the semantic categories. Finally, the word is assigned to the category with the closest centroid (smallest cosine distance).

The supervised learning algorithms have been used from a package called *liblinear*. Liblinear is a software package for performing linear-classifier learning (binary, multi-class, and regression) (Fan et al., 2008). Liblinear supports various training methods and objectives, such as SVM and LR. From the liblinear package, linear SVM and LR have been used to train a classifier which classifies the words into their semantic category, by using the words vector representation as an input directly. To use LR and SVM in liblinear library, it is a matter of changing $-s$ parameter value. If $-s$ is 0, we are using *L2-regularized logistic regression*. The default value of $-s$ is 1 which is *L2-regularized L2-loss support vector classification (dual)*. The default value is the one that we have been using to train the classifiers on SVM. Every step and procedure that have been followed for both SVM and LR, are the same in the liblinear library.

The hyperparameters of the models have been tuned using a grid search script from LIBSVM. LIBSVM is another library for SVM which contains gird search script to search the value of the right parameter to build a classifier (Chang and Lin, 2011). However, LIBSVM is not as efficient as Liblinear when it comes to large-scale problems. Liblinear is efficient for training, large-scale problems (Fan et al., 2008). In our study, the number of features is 17,400. Therefore, it is efficient to use liblinear when the number of features is very large (Hsu, Chang, and Lin, 2003).

The LIBSVM grid search script has been used to find the best C parameter value. A number in between the range 0 and 20 in step 0.05 have been tuned. The following command has been used to tune the C parameter value:
./grid.py -log2c 0,20,0.05 -log2g null -v 10 -svmtrain ./train Eurovoc.train.
In the command, *-log2g null* shows that liblinear does not support the gamma parameter. *-v 10* shows that ten-fold cross-validation has applied to tune the parameter. The grid search script uses the cross-validation technique to estimate the accuracy of each parameter's combination in the specified range and helps to decide the best parameters value. Therefore, ten-fold cross-validation has been conducted in order to select the best parameter C. Then we train a classifier on the training set using the best C parameter value and predict the test set. For all experiments, ten-fold stratified cross-validation has been used on the datasets to train the classifier models. Thus, ten training and ten test datasets have generated. Stratified cross-validation is explained in Chapter 3 Section 3.3.3. In our study, 10% of the data has considered for the test and 90% of the data for the training set. These training and test sets have been scaled in between the range [0; 1] before tuning, training, and testing. The main advantage of scaling is to avoid features in greater numeric ranges dominating those in smaller numeric ranges (Hsu, Chang, and Lin, 2003).

The other main classification method is a multi-relational matrix factorization (MRMF). Basically, matrix factorization focuses on only one relation of entities. However, entities may have more than one relationship which can be used for deep analysis and training of entities. Thus, the main idea of MRMF is to use the multi relations of entities. In our study, we have considered multiple relationship types to build a model which classify semantically related words in their right semantic category using MRMF. These relations are the target word with distributional information, lexical information and semantic categories information. In our case, semantic categories information is the class/label information of the words from their dataset (SC53 and Eurovoc). MRMF can thus easily integrate these sources of information and train a classifier. Once MRMF integrates the different information, it generates matrices. Then, the generated matrices can be folded in with the test matrix in order

to classify the unknown words with a given formula. The technical detail explanation of MRMF is in Section 4.3.

The idea of integrating different sources of information has been applied to SVM and LR as well. Lexical and distributional information are the main sources of information which have been considered to be integrated by SVM, LR and MRMF methods. Integrating the two sources of information using MRMF method is straightforward and easy. However, the ensemble classifier is the obvious alternative to integrating the two sources of information using the SVM method. An ensemble classifier uses the results of the classifiers using only one type of information. As a result, the ensemble classifier has been trained on the results of the SVM using WordNet and distributional features. In addition, the LR classifier has also been used to integrate the sources of information via the ensemble method. LR gives probabilities for each class and selects the class with the highest probability to classify a word. The ensemble classifier can use the probabilities for each class. Though we expect that LR to be inferior to SVM because it might have an advantage to use the class's probabilities in an ensemble classifier.

4.3 Multi-Relational Matrix Factorization

MRMF is introduced by Lippert et al. (2008) for relation prediction in multi-relational domains using matrix factorization. The novel idea came to be able to deal with an arbitrary number of relation types and entity types in a domain of interaction to build a generalized model. Instead of using a model with a single relation type between two entity types (Lippert et al., 2008). The approach prediction performance investigated on user-movie, and gene function datasets as explained in introduction Chapter 1 Section 1.2. Weighted MRMF which has been used in this study as well, is defined by Drumond et al. (2014). The idea of weighted MRMF is to build a model which optimize each target relation individually, instead of using the same set of parameters to make predictions for all target relations. Drumond et al. (2014) introduced specific parameters for each target by coupling them with a set of shared auxiliary parameters, which has a regularizing effect on the target specific ones. The state of the art approaches did not differentiate the target and auxiliary relations. Therefore, they use the same parameters for predicting all the targets. This means that the learned parameters are a compromise for the performance overall targets, but not for each specific one (Drumond et al., 2014). To clear the idea of the target and auxiliary relations in our case, we have used Figure 4.1. We have used the same approach that Drumond et al. (2014) have also used on social network data to explain the meaning of target and auxiliary relations.

In our study, three different matrices which represent the relations of the words have been used. These are: 1. the matrix of words and semantic category features, 2. the matrix of words and context features, and 3. the matrix of words and WordNet features. The following subsections explain how MRMF uses and integrates the relations to build a model which classify semantically-related words. Section 4.3.1 explains how MRMF use only the semantic categories and distributional features information of a word and build a model without the lexical information. Section 4.3.2 explains the learning algorithms which are used to optimize the objective function and build the classifier using the above two relations. In addition, the section explains the prediction equations that have been used to classify new words. Section 4.3.3 explains how MRMF integrates the three relations to build a classifier. In addition, this section discusses the algorithm which is used to optimize the objective

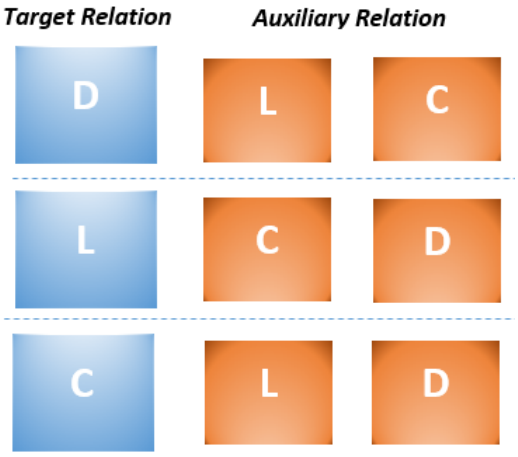

FIGURE 4.1: There are three relations: distributional information **D**, lexical information **L** and the words given semantic categories **C**, between two entities *words* and *semantic categories*. This example shows the corresponding three cases for which each relation is acting as a target and the rest ones as an auxiliary.

function, and the prediction equations. Finally, section 4.3.4 discusses the parameters which have been tuned to build a classifier.

4.3.1 MRMF on two Matrices

The MRMF method has been studied by the first two matrices/relations, which are word and context features matrix X and words and semantic category features matrix Y. Let's assume the following problem to elaborate the method study on these relations. We have:

- m words;
- n features for each word (e.g. positive pointwise mutual information (PPMI) values based on the co-occurrence data);
- c semantic categories;

The first matrix is, $X \in \mathbb{R}^{m \times n}$ where each row of X represents the feature vector of a word. The second matrix is, $Y \in \{0,1\}^{m \times c}$. $Y_{i,j}$ has value 1 if the word i belongs to the category c, otherwise, it has value 0. The MRMF method with the two matrices is referred to as $MRMF\_2M$.

The idea of matrix factorization is that X can be approximated by the product of two smaller matrices U and V, where $U \in \mathbb{R}^{m \times k}$ is a matrix of words and latent features and $V \in \mathbb{R}^{n \times k}$ is a matrix of context features (context words) and the same latent features. The latent features size k can be chosen freely with in $k << n$. The second matrix, Y, can be decomposed in the same way. Y can be approximated by the product of two smaller matrices U and C, where $U \in \mathbb{R}^{m \times k}$ is a matrix of words and latent features, and $C \in \mathbb{R}^{c \times k}$ is a matrix of semantic categories and the same latent features.

The idea of MRMF is that both decompositions of X and Y use the same factor matrix U of words and latent features size, thus the latent features form the link between the context features (context words) and the categories in this case. The matrices U, V and C are constructed using the training data.

More formally, X and Y can be factorized as follows:

$$X \approx UV^T \tag{4.2}$$

$$Y \approx UC^T \tag{4.3}$$

The overall decomposition of the two matrices for MRMF method is visualized in Figure 4.2.

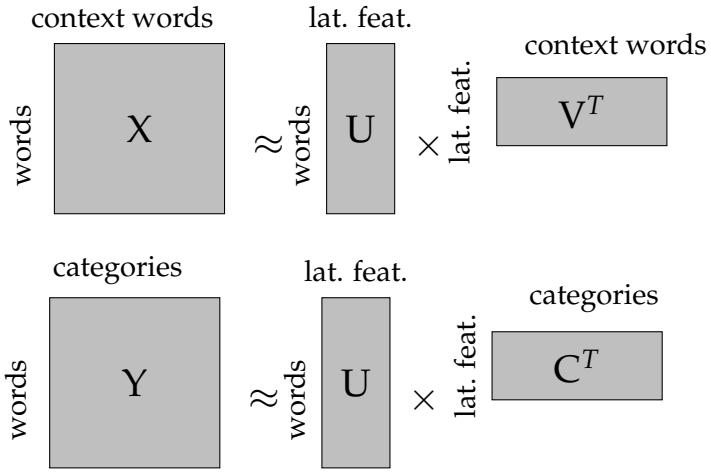

FIGURE 4.2: Visual overview of the matrix decomposition used for semantic categorization on two matrices

The problem is now to minimize an objective function with respect to L2 loss function. The objective function is shown in Equation 4.4.

$$\begin{aligned} \underset{U,V,C}{\arg\min} \quad & \alpha_X \frac{1}{2}||X - UV^T||_F^2 + \alpha_Y \frac{1}{2}||Y - UC^T||_F^2 \\ & + \frac{\lambda_U}{2}||U||_F^2 + \frac{\lambda_V}{2}||V||_F^2 + \frac{\lambda_C}{2}||C||_F^2 \end{aligned} \tag{4.4}$$

The L2 loss function is basically minimizing the sum of the square of the differences between the target value and the estimated values.

4.3.2 Learning Algorithm and Predictions

One of the optimization algorithms that most often used is block coordinate descent. This algorithm optimizes the objective function through a sequence of one-dimensional optimization. This means that it solves the optimization problems by minimizing it along one direction at a time in a loop (Wright, 2015). The python script of MRMF on the coordinate descent algorithm that we used for our study, is given in Appendix A.

When we use the algorithm in our study, first U, V, and C are initialized with random values. Then, the minimization problem is solved for each one of the matrices. This is repeated until convergence. The algorithm is shown in Algorithm 1. The algorithm optimizes the objective function which is shown in Equation 4.4.

Finally, Equation 4.5 is used for a set of new words X^{test} to predict their semantic categories.

$$Y^{\text{test}} \approx U^{\text{test}} C^T \tag{4.5}$$

However, U^{test} is unknown. The standard way to estimate U^{test} is through a fold-in:

$$U^{\text{test}} = \arg\min_{\hat{U}} ||X^{\text{test}} - \hat{U}V^T||_F^2 \tag{4.6}$$

$$U^{\text{test}} = X^{\text{test}} V (V^T V)^{-1} \tag{4.7}$$

Algorithm 1 Block coordinate descent optimization algorithm for L2-MRMF_2M

procedure MRMF-COORDINATE DESCENT
input: X, Y, k, *weight constants* α_X, α_Y, *regularization constants* $\lambda_U, \lambda_V, \lambda_C$
  $U \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $V \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $C \sim \mathcal{N}(0, \sigma\mathbf{I})$
  repeat
    $U \leftarrow (\alpha_X XV + \alpha_Y YC)\left(\alpha_X V^T V + \alpha_Y C^T C + \lambda_U \mathbf{I}\right)^{-1}$
    $V \leftarrow \left(\left(\alpha_X U^T U + \lambda_V \mathbf{I}\right)^{-1} \alpha_X U^T X\right)^T$
    $C \leftarrow \left(\left(\alpha_Y U^T U + \lambda_C \mathbf{I}\right)^{-1} \alpha_Y U^T Y\right)^T$
  until convergence
  return U, V, C
end procedure

4.3.3 MRMF on three Matrices

The MRMF method allows one to elegantly integrate many different sources of information. In this experiment, the lexical and distributional information has been integrated by extending the MRMF method that has described in Section 4.3.1.

Matrices X and Y that we use in this section, are the same matrices that we have seen in section 4.3.1. Matrix Z is newly-added matrix. It contains lexical information of the words which has extracted from WordNet by considering hypernym relation. X, Y, and Z can be factorized more formally as follows:

$$X \approx UV^T \tag{4.8}$$

$$Y \approx UC^T \tag{4.9}$$

$$Z \approx UB^T \tag{4.10}$$

The overall decomposition of the three matrices is visualized in Figure 4.3. This MRMF method is referred to as *MRMF_3M*. This method uses the same algorithm, the coordinate decent algorithm. However, it has modified to fit the third matrix Z information in the algorithm. The modified coordinate descent algorithm is shown in Algorithm 2. This algorithm also optimizes the objective function with respects to L2 loss function. The objective function and the fold-in equations have also been modified to include the Z matrix information as shown in Equation 4.11 and 4.12

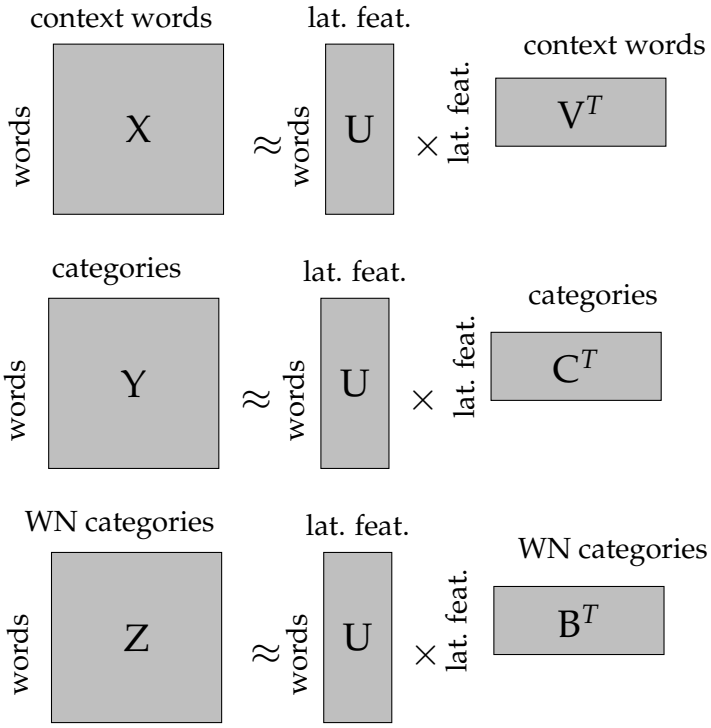

FIGURE 4.3: Visual overview of the matrix decomposition used for semantic categorization on three matrices.

respectively. But, we can still use Equation 4.5 to predict new words, once we have the U^{test} using Equation 4.12.

$$\begin{aligned}\underset{U,V,B,C}{\arg\min}\quad & \alpha_X \frac{1}{2}||X - UV^T||_F^2 + \alpha_Z \frac{1}{2}||Z - UB^T||_F^2 + \alpha_Y \frac{1}{2}||Y - UC^T||_F^2 \\ & + \frac{\lambda_U}{2}||U||_F^2 + \frac{\lambda_V}{2}||V||_F^2 + \frac{\lambda_C}{2}||B||_F^2 + \frac{\lambda_C}{2}||C||_F^2\end{aligned} \tag{4.11}$$

$$U^{\text{test}} = X^{\text{test}} V (V^T V)^{-1} + Z^{\text{test}} B (B^T B)^{-1} \tag{4.12}$$

4.3.4 Parameter Selection

A combination of parameters has been trained with a wide range of values to find the best parameters value setting. The parameters are the learning rate, latent features, and regularization parameters.

For MRMF_2M method, the parameters that have been trained, are the learning rate α_x and the regularization parameters λ_u, λ_v and λ_c . For SC53 dataset, a range between $\frac{1}{\#\_SC53\_instances}$ and $1 \cdot 10^{-7}$ has been considered to find the best learning rate α_x the parameter value, and a range between $\frac{1}{\#\_Eurovoc\_instances}$ and $1 \cdot 10^{-7}$ for the Eurovoc dataset. The learning rate of α_y has been set to 1; Because we are building the model to classify the words to the Y matrix semantic categories. Therefore, all information on Y training data is needed to train the model. The regularization constants λ_u, λ_v and λ_c have used the same range of value which relies on a range

Algorithm 2 Block coordinate descent optimization algorithm for L2-MRMF_3M

procedure MRMF-COORDINATE DESCENT
input: X, Z, Y, k, *weight constants* $\alpha_X, \alpha_Y, \alpha_Z$, *regularization constants* $\lambda_U, \lambda_V, \lambda_B, \lambda_C$
  $U \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $V \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $B \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $C \sim \mathcal{N}(0, \sigma\mathbf{I})$
  repeat
    $U \leftarrow \left(\alpha_X XV + \alpha_Y YC + \alpha_Z ZB\right)\left(\alpha_X V^T V + \alpha_Y C^T C + \alpha_Z B^T B + \lambda_U \mathbf{I}\right)^{-1}$
    $V \leftarrow \left(\left(\alpha_X U^T U + \lambda_V \mathbf{I}\right)^{-1} \alpha_X U^T X\right)^T$
    $B \leftarrow \left(\left(\alpha_Z U^T U + \lambda_B \mathbf{I}\right)^{-1} \alpha_Z U^T Z\right)^T$
    $C \leftarrow \left(\left(\alpha_Y U^T U + \lambda_C \mathbf{I}\right)^{-1} \alpha_Y U^T Y\right)^T$
  until convergence
  return U, V, B, C
end procedure

between $1 \cdot 10^{-2}$ and $1 \cdot 10^{-10}$. For the latent features k, we have considered a range between 50 and 200.

Figure 4.4 and Figure 4.5 show the parameters α_x and k value training to find the best value for the SC53 and Eurovoc datasets respectively. As both figures show, the k parameter gives a high accuracy on value 200 and goes flat after that. In method MRMF_2, the k parameter has been performed better with the α_x parameter value around 0.002 for SC53 dataset and parameter values in between 0.0003 and 0.0001 for the Eurovoc dataset.

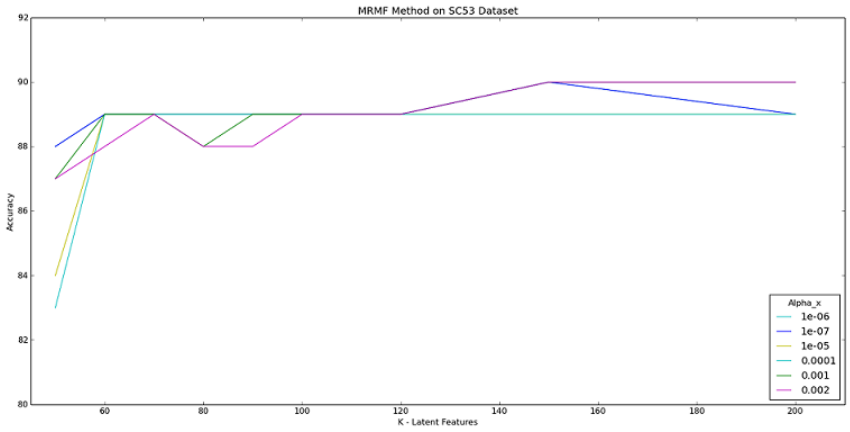

FIGURE 4.4: Accuracy of MRMF_2M with different k and α_x parameter values on the SC53 dataset

The parameters that have been trained for MRMF_3M method, are learning rate α_x and α_z, and regularization parameters λ_u, λ_v and λ_b. For SC53 dataset, a range between $\frac{1}{\#\_SC53\_instances}$ and $1 \cdot 10^{-7}$ has been considered to find the best learning rate α_x and α_z parameters value, and a range between $\frac{1}{\#\_Eurovoc\_instances}$ and $1 \cdot 10^{-7}$ for the Eurovoc dataset. Here again, the learning rate of α_y has been set to 1 for the same reason which is given above in MRMF_2M method parameter selection. The regularization constants λ_u, λ_v, λ_c and λ_b and latent features size k have also

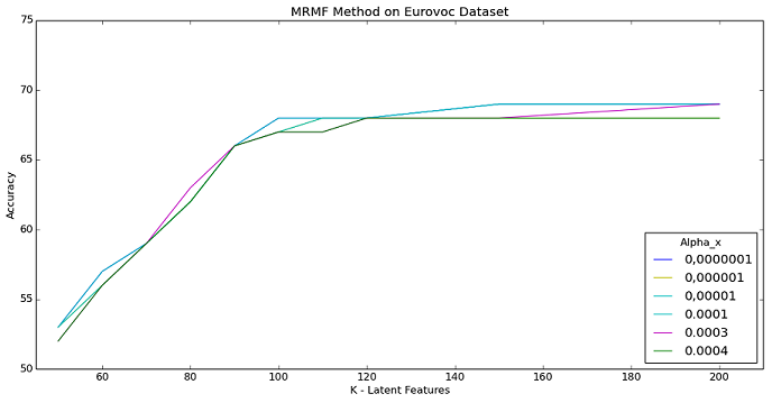

FIGURE 4.5: Accuracy of MRMF_2M with different k and α_x parameter values on the Eurovoc dataset

used the same range of value as explained above in MRMF_2M method parameter selection. Figure 4.6 shows the parameters α_x, α_z and k value training to find the best value for Eurovoc dataset as an example. As the figure shows, the k parameter gives high accuracy at value 200 and plateaus after that. For the SC53 dataset, the k parameter has performed better when α_x is 0.001 and α_z is 0.0004. For the Eurovoc dataset, when α_x is $1 \cdot 10^{-6}$ and α_z is $1 \cdot 10^{-8}$, MRMF_3 gives optimal results with k value 200 as Figure 4.6 shows.

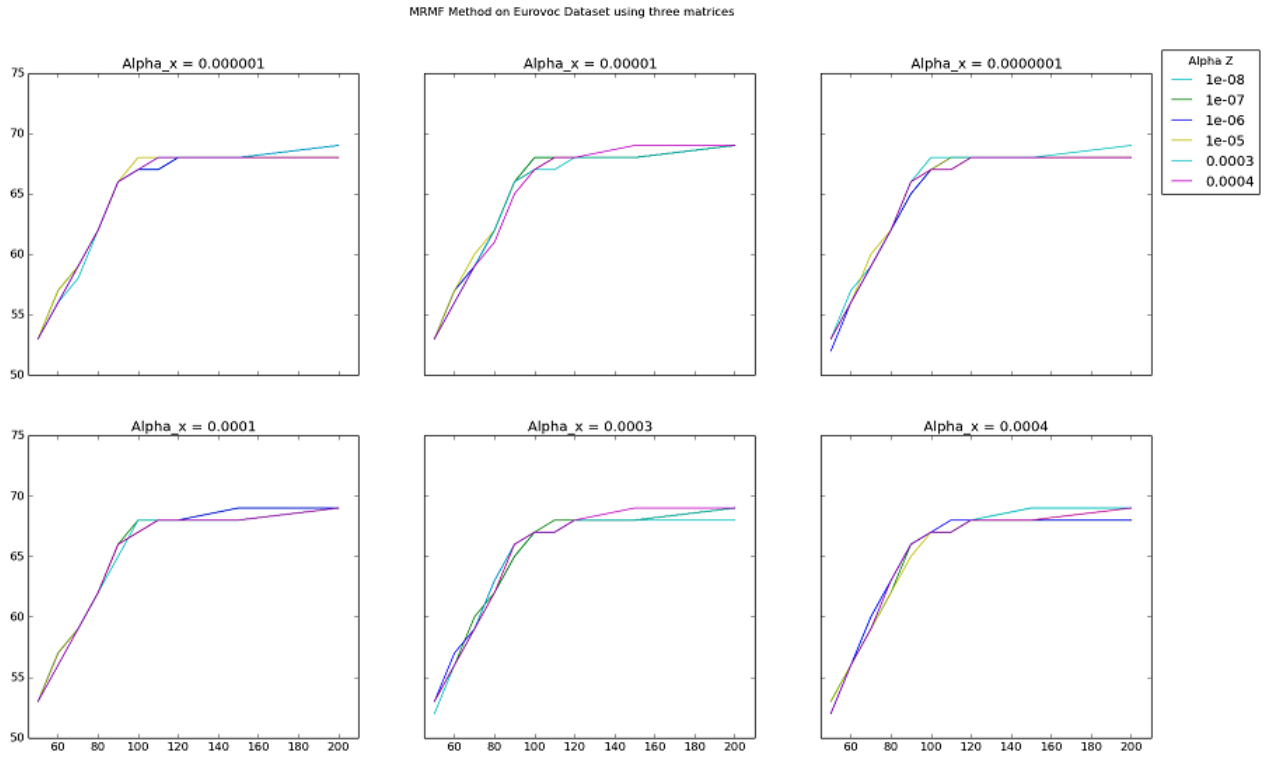

FIGURE 4.6: Accuracy of MRMF_3M with different k, α_x and α_z parameter values of the Eurovoc dataset

4.4 Evaluation

For the purposes of evaluation, ten-fold cross-validation has been used. For all experiments, the same stratified split has been applied. This is basically the same as the leave–one–out setup that is used by Bullinaria and Levy (2012) when the number of items in the categories is the same. Leave–one–out setup is used in one of our method *nearest centroid* as well which is explained in Section 4.2.4. However, in a leave–one–out experiment, an intelligent classifier eventually might learn that always an element to be classified to the smallest class when the dataset does not has equal size classes. This problem is avoided by using stratified cross-validation. Stratified cross-validation has explained with an example in Chapter 3 Section 3.3.3. In ten-cross-validation, 10% of the data is test set and 90% of the data is the training set. In this study, all methods have been using ten-fold cross-validation to train and evaluate classifiers.

4.5 Result

The performance of each classifier on each dataset has summarized in the result Table 4.2 with their standard error (Stand. Err.). In the result table, the sources of information which have used to represent the words have abbreviated. DF and WN represent distributional features and WordNet (hypernym) information respectively. The best accuracy, that Bullinaria and Levy (2012) reached for the SC53 dataset (using the nearest centroid classifier), is $0,86$. This result is reproduced in our study using roughly the same parameter choices and settings with Bullinaria and Levy (2012) to construct the context vector word representation. Applying the same method (nearest centroid) on the Eurovoc dataset gives an accuracy of 0,58. The hypernym features (WN) have only been used in the supervised learning methods.

As the result table shows that the supervised learning classifiers (LR, SVM, and MRMF) are superior to the unsupervised learning (nearest centroid classifier) on both datasets, except the classifiers which have been trained only using WN information. All classifiers which have been trained on WN stayed behind from the whole classifiers. This shows that WN cannot represent words as much as DF represents. The WN representation has added in the study just to be integrated with DF and improve the performance of the classifiers. As expected, the classifiers which have been trained by the integrated information of WN and DF have improved their performance.

The supervised classifiers which have been using only DF, have shown the same performance between each other, except the LR classifier on the Eurovoc dataset which has stayed behind. For the SC53 dataset, both the supervised classifiers using only DF and the classifiers using a combination of DF and WN have outperformed the best results which have been reported up to now. Keith, Westbury, and Goldman (2015) reported an accuracy of $0,96$ when they reproduced the experiment of Bullinaria and Levy. However, their result is not comparable to ours, since they used only a part of the data for evaluation. In other words, they did not test all instance as training and test sets by using an evaluation method such as cross-validation.

We can see that the integration of lexical and distributional information using MRMF clearly improves the result for both datasets. The ensemble methods have also improved the results, but they have stayed behind comparing the result of MRMF. Since the LR results for the Eurovoc dataset have stayed much behind the

SVM and MRMF results, we did not test the ensemble method on this dataset. As an additional reason, the ensemble method did not perform well in the SC53 dataset.

If we investigate the SC53 dataset words which have been predicted by the MRMF_3M classifier, there is still a small number of real errors. E.g. the word *mixer* has been classified as a non-alcoholic beverage and *nun* as a relative. However, most errors are not real errors, for instance, the word *foot* that has been classified as a body part by MRMF and is a unit of distance in the dataset. A *knife* has classified as a weapon instead of a kitchen utensil; *shoes* as a type of footwear instead of clothing; and a *bass* as a musical instrument instead of a fish.

Given the type of errors that are made, we can conclude that the SC53 dataset has reached the highest possible level of accuracy. The Eurovoc dataset clearly is much harder and has still room for improvement.

	Methods	**Eurovoc**	**Stand. Err.**	**SC53**	**Stand. Err.**
	Nearest Centroid (reported)	-	-	0,86 [3]	-
	Nearest Centroid (reproduced)	0,58	0,14	0,86	0,04
WN	LR	0,45	0,17	0,74	0,11
	SVM	0,50	0,16	0,73	0,10
	MRMF_2M	0,48	0,17	0,79	0,07
DF	LR	0,56	0,16	0,90	0,04
	SVM	0,69	0,10	0,90	0,03
	MRMF_2M	0,69	0,10	0,90	0,03
DF + WN	MRMF_3M	**0,71**	0,10	**0,93**	0,02
	2xLR + SVM	-	-	0,89	0,03
	2xSVM+ SVM	-	-	0,92	0,02

TABLE 4.2: Accuracy of classification on Eurovoc and SC53 datasets. Results are averages from ten-fold cross-validation. The ensemble method on Eurovoc dataset did not compute, because, the LR results for the Eurovoc dataset have stayed far behind the SVM and MRMF results. In addition, the ensemble method did not perform well in the SC53 dataset.

4.6 Conclusion

This work has investigated the supervised learning approach for semantic classification of words task in distributional semantics. In the study, the classification which has trained on a supervised approach has outperformed the unsupervised learning approach (i.e. a distance based classifier [nearest centroid]). This study has been shown in two different datasets SC53 and Eurovoc. SC53 is commonly used for the semantic classification task. In both datasets, the classifiers which have trained on SVM and MRMF using distributional features information as an input, have performed better than the rest of the classifiers. For SC53 dataset, we have obtained results that are beyond the state of the art.

We have compiled the Eurovoc dataset to make the classification task closer to real applications with more semantic categories. This data set is clearly much harder, but experiments on this dataset confirm all conclusions from the experiment on the smaller dataset of SC53.

In order to improve the results, we finally have investigated the possibility to include information from WordNet. While an ensemble classifier was not very suc-

cessful in combining the two sources of information, MRMF has been able to integrate the two types of information and improve the results significantly.

Chapter 5

Identification of Semantic Relations

5.1 Introduction

In distributional semantics, words are represented by large number of context features as explained in Chapter 2. In most cases, context features are based on co-occurrence numbers or probabilities with other words. Words are semantically related when they have similar or related co-occurrence vectors. The relatedness of the co-occurrence vectors can be found by computing their distance using distance measures such as the cosine and Euclidean distance. A simple approach which decides whether two words are semantically related or not, can be based on using the similarity of their associated vectors, directly, as an input. This approach trains a model on supervised learning algorithms using the words associated vector directly. We use the model to predict the relatedness of the words. In our study (Chapter 3), we have shown the performance of this approach. In this chapter, we call this approach *HsH-Supervised* system. Then, the system has used on a shared task challenge that we have participated. This chapter explains the shared task challenge and discusses the proposed solutions.

This study was initiated by CogALex-V 2016[1]. CogALex-V 2016 has proposed a shared task on the corpus-based identification of semantic relations in a pair of words.The objective of the task is to find a better method which classifies semantically-related pairs of words. This shared task is more related to one of our previous work which has presented in Chapter 3. However, our previous study, main objective is to investigate the supervised learning methods over the unsupervised learning methods on classifying semantically related and not related pair of words task. Then, supervised learning methods have proposed for this task. Specifically, *HsH-Supervised* system was proposed. For the shared task, this system has been considered as one of the methods.

The shared task has two sub-tasks. The first sub-task is to decide whether a pair of words are semantically related or not. The second sub-task is to decide which kind of semantic relation that the pair of words holds. Four semantic relations have been proposed for the second sub-task. The relations are *Synonymy*, *Antonymy*, *Hypernymy* and *Meronymy*. The remaining pairs which are not under the category of these four relations have been labeled as *random*.

The shared task has carried out on two phases of study. The first phase is to study whether or not HsH-Supervised can be the one for the given shared task. The second phase is to investigate the performance of multi-relational matrix factorization (MRMF) method on the given shared task. The MRMF method has found as a novel

[1] https://sites.google.com/site/cogalex2016/home/shared-task

method (in distributional semantics) in our semantic classification study (Chapter 4). The second phase came after the publication of the first phase. MRMF method is motivated on this phase, because of its positive performance on semantic classification task. Therefore, it has been studied on classifying semantically-related pair of words tasks as well.

Basically, in the first phase, the supervised learning method that has been proposed in our semantic similarity study Chapter (3) has been used for the challenge as well. This method has shown a better performance when classifying semantically-related pair of words compared to the state of the art methods in our previous study. The main idea of the method is, first, to represent the pair of words by a vector, and train a classifier on a supervised learning algorithm. The vector which represents the pair of words is constructed by using simple mathematical operations. Specifically, in our previous study, we have proposed the use of pairwise multiplication operations on the context vectors of each word of the pair to construct the vector representation. Then a model is learned by using the pairs vector representation as an input for supervised algorithms such as SVM. Now, the similarity between two words can be learned via a supervised classification method. Besides our study, a number of papers have also been proposed to use derived distributional features to represent each pair of words by a large distributional feature vector, e.g. (Hagiwara, 2008). These research pieces have explained in Chapter 2 Section 2.2.

The second phase has initiated mainly because of the rank which is given for the first phase method. After the submission of all systems, they have ranked. *HsH-Supervised* is the system which has proposed in the first phase for the shared task. This system has ranked 5th from the number of six participants on the challenge. Therefore, the second phase study came for two main reasons. These are: 1. to study another better method for the given shared task, 2. to investigate the performance of MRMF on semantic similarity task. In general, the approach of the second phase is to use MRMF method on the given semantic similarity shared task. MRMF has shown good performance on semantic classification task as explained in Chapter 4. This method has even outperformed the state of the art methods and other proposed supervised learning methods such as SVM to classify semantically related words. Therefore, it was expected to perform better than *HsH-Supervised* and also the other top-ranked systems by the participants in the challenge. MRMF has the capacity to integrate different sources of information and relations easily with a better quality comparing other methods such as ensemble method as explained in Chapter 4. As Chapter 4 explains that integrating different relations of entities is better than single relation in order to train a model.

The general overview of this chapter is studying the HsH-Supervised and MRMF method on semantic similarity task. Basically, HsH-Supervised is already studied on semantic similarity task in Chapter 3. Here, it simply has proposed for the shared task challenge participation. MRMF has studied on semantic similarity task for the first time. The chapter focuses on investigating the performance of MRMF method on semantic similarity task, after explaining the HsH-Supervised system performance on the given shared task.

The rest of the paper is organized as follows. Section 5.2 discusses the CogALex-V shared task with the given datasets. Section 5.3 explains the two phases methods. Finally, results and conclusion have been discussed in Section 5.4 and in Section 5.5 respectively.

5.2 CogALex-V task

The CogALex-V shared task has provided two datasets that contain a pair of words along with a classification of their semantic relation. Each sub-task of the challenge has their own datasets. The pair of words that the two datasets contain is the same but their semantic relation label is different. The first given dataset for the first sub-task distinguishes the pair of words only between related and unrelated relation, while the second dataset distinguishes in four types of semantic relations for the second sub-task. The four types of relations between the related pair of words are; *synonymy*, *antonymy*, *hypernymy*, and *meronymy*. In the second task, the not-related pairs labeled as *random*. The definition of each relation has shown below: *Notations: Pairs (A: word 1, B: word 2)*:

- Synonymy (SYN): B can be used with the same meaning as A
- Antonymy (ANT): B can be used as the opposite of A
- Hypernymy (HYPER): A is a kind of B
- Meronymy (PART_OF): A is a part of B
- RANDOM (if there is no semantic relation between A and B)

The sample data of the first and second sub-tasks datasets have shown in Figure 5.1 and Figure 5.2 respectively.

A	B	Relation
coach	teach	TRUE
coach	teacher	TRUE
coach	coat	FALSE
dance	move	TRUE
dance	human	FALSE
dance	idea	FALSE
follow	lead	TRUE
follow	understand	TRUE
follow	exchange	FALSE
father	parent	TRUE

TABLE 5.1: Sample data of the first Sub-task dataset

Both sub-tasks training and test data contain 3054 and 4260 pairs of words, respectively. From the 3054 training pairs, 826 pairs are semantically related, the remaining 2228 pairs are not related. From the 4260 test pairs, 1201 of them are related and 3059 are not related. In contrast to many other related datasets such as *Eurovoc-0, Eurovoc-1, Eurovoc-2*, the words in these set are very heterogeneous. The sets contain *nouns*, *adjectives*, *verbs* and even *pronouns*. For example, pairs like *arm–leg* or *vegetable-meat* are considered as antonyms and hence related words, while other pairs that are related somewhat more indirectly, such as *breast-leg*, *vegetable-apple* or *run-athlete* are classified as unrelated. The *breast-leg* and *vegetable-apple* pars have a co-meronyms relation. Athletics could be considered as a hypernym of running, therefore, *run-athlete* pair could have been labeled as related. Thus it becomes clear that the dataset is far from trivial and is a real challenge for automatic classification.

A	B	Relation
coach	teach	HYPER
dance	move	HYPER
everything	all	SYN
division	split	SYN
coach	coat	RANDOM
dance	human	RANDOM
follow	lead	ANT
woman	male	ANT
father	family	PART_OF
string	kite	PART_OF

TABLE 5.2: Sample data of the second sub-task dataset

5.3 Methodology

This section explains the methods which have been proposed for the given shared task. The first method is a system which is called *HsH-Supervised* that has proposed for the semantic similarity task in our previous study. HsH-Supervised learns a model on SVM by using the pair of words vector representation as an input. The pair of words vector representation has been constructed by applying pairwise multiplication (Hadamard-Product) on the context vectors of the words. The second method is MRMF. MRMF also uses the context vectors of each word of the pair to train a model.

In order to study the methods, first, each word has been represented by context vectors, since both methods use context vectors of each word. These context vectors have been constructed by following the exact procedure which has been explained in Chapter 2 Section 2.5.1. Then point-wise mutual information (PPMI) matrix has constructed which contains each word context vector representation. PPMI is a vector weighting which has applied on the context vectors after constructing them from the ukWaC corpus by learning some parameter values. Then each word has been represented by distributed features with 17,400 vector size. The complete explanation of the PPMI matrix construction has been explained in Chapter 2 Section 2.5.1. The following subsections will explain the two methods of study in detail.

5.3.1 HsH-Supervised System

HsH-Supervised uses a pair of words vector representation as an input to train a classifier on classification algorithms. Thus, first, it represents a pair of words by a vector which has been computed by applying mathematical operations on the pair of words' context vectors. Specifically, HsH-Supervised uses pairwise multiplication operation to construct the pairs vector representation. Pairwise multiplication has shown to give good results in our previous study (Chapter 3). As an additional method, the *addition* method has also included to constructed the pairs vector representation. The *addition* method is one of the methods which has a comparable performance with multiplication methods in our semantic similarity study (Chapter 3). Thus, it has been considered in this study to compare it with a HsH-Supervised system which uses the multiplication method to represent pairs by a vector. The mathematical operation of pairwise multiplication and addition are shown in Equation 5.1 and Equation 5.2. The equation includes the following annotations: $\overrightarrow{A}$ and

$\overrightarrow{B}$ represent each context vector of the words of the pair; $\overrightarrow{A} = \langle a_1, a_2, a_3, ..., a_n \rangle$ and $\overrightarrow{B} = \langle b_1, b_2, b_3, ..., b_n \rangle$.

$$\begin{aligned} \overrightarrow{V} &= \overrightarrow{A} \odot \overrightarrow{B} \\ &= (a_1 \cdot b_1, a_2 \cdot b_2, a_3 \cdot b_3, ..., a_n \cdot b_n) \end{aligned} \tag{5.1}$$

$$\begin{aligned} \overrightarrow{V} &= \overrightarrow{A} + \overrightarrow{B} \\ &= (a_1 + b_1, a_2 + b_2, a_3 + b_3, ..., a_n + b_n) \end{aligned} \tag{5.2}$$

The above equations have been applied on the vectors which are the unit vector of the context vectors. A unit vector is any vector with a magnitude length of one. As it has been explained in Chapter 3, the unit vector has improved the accuracy of the classifier when it is applied to the context vectors before computing the mathematical operations. Thus, here, it has applied on the context vectors before the addition and multiplication operations as well. To find a unit vector of a vector, it is simply dividing the vector by the magnitude of the vector as shown in Equation 5.3.

$$\hat{u} = \frac{\overrightarrow{v}}{|v|} \tag{5.3}$$

In mathematics, magnitude represents the size of a mathematical object. The magnitude equation has been shown as follows in Equation 5.4.

$$|v| = \sqrt{\sum_{i=0}^{n} {a_i}^2} \tag{5.4}$$

After the pair of words vector representation, linear SVM from the liblinear package has been used to learn a model and classify the pair of words which have been represented by a single vector. The hyperparameters of the models have been tuned using grid search from LIBSVM on the training data. The gird search script has been used to find the best C parameter value. To find the best C parameter value, the numbers in between the range -5 $\leq$ log2C $\leq$ 15 have been tested in step 0.05. The optimal values have been found using ten-fold cross-validation on the training data. The gird search script uses cross-validation technique to tune the parameter values. The right selection of the hyper-parameter values should minimize the risk of overfitting.

Besides the given baseline by the shared task, we have been considering the classical cosine similarity output results as well as a baseline. Cosine computes the distance between the context vectors of the two words. The cosine distance measure needs an optimal split value to split or classify semantically related and unrelated pairs. Then, the optimal split value has learned on the training data. Then, optimal value 0.0842 has found for classification or split. Thus for test data, the optimal value considers pairs with context vectors that have a cosine distance above or equal 0.0842 to be semantically-related, otherwise unrelated.

As a further simple baseline for the first sub-task, a classifier that considers each pair as semantically related has been used. In fact, this is a type of majority classifier which always assigns the largest *evaluated* category. For the second sub-task, the largest evaluated category in the training data is the hypernym relation (255 pairs). Thus, this classifier assigns a hypernym to each pair.

5.3.2 MRMF Method

The MRMF method has shown good performance on semantic classification task as it has been explained in detail in Chapter 4. In Chapter 4, the idea of matrix factorization and MRMF method has been explained in detail. Therefore, this chapter focuses only on how this method has been applied to the given shared semantic similarity task. The method has considered two approaches to building a classifier for the task.

The first approach is to build a classifier by integrating the distributional information of each word of the pair and the pair of words relation information. Then, these two pieces of information have been represented by two matrices, X and Y. This approach overview is shown in Figure 5.1. Let's assume the following problem for this approach and explain the matrices. We have:

- m words;
- n features for each word (i.e. positive pointwise mutual information (PPMI) values based on the co-occurrence data);

The distributional information is represented by a matrix $X \in \mathbb{R}^{m \times n}$ where each row of X represents the distributional features of a word. X can be approximated by the product of two smaller matrices U and V, where U is a matrix of words and latent features $U \in \mathbb{R}^{m \times k}$, and V is a matrix of distributional features and the same latent features $V \in \mathbb{R}^{n \times k}$. The latent features size k can be chosen freely within $k << n$.

Matrix Y represents the semantic relation between each word of the pair. The matrix can be formulated as follow: $Y \in \{0,1\}^{m \times m}$. $Y_{i,j}$ has value 1 if the first word of the pair i is semantically related with the second word j, otherwise 0. Y can be approximated by the product of matrix U and the transpose of matrix U^T, where U is a matrix of words.

More formally, X and Y can be factorized as follows:

$$X \approx UV^T \tag{5.5}$$

$$Y \approx UU^T \tag{5.6}$$

The problem is now to minimize the objective function with respect to L2 loss function. The objective function is shown in Equation 5.10.

Like the first approach, the second approach is also building a classifier by integrating different information. In this approach, the distributional information of the first word of the pair, the second word of the pair and the semantic relation of the pair of the words have been integrated to build the classifier. For these three pieces of information, three different matrices (X, Y and Z) have been prepared. This approach overview has been shown in Figure 5.2. Let's assume the following problem for this approach and explain the matrices: We have

- $m1$ first word of the pair;
- $m2$ second word of the pair;
- n features for each word (i.e. positive pointwise mutual information (PPMI) values based on the co-occurrence data);

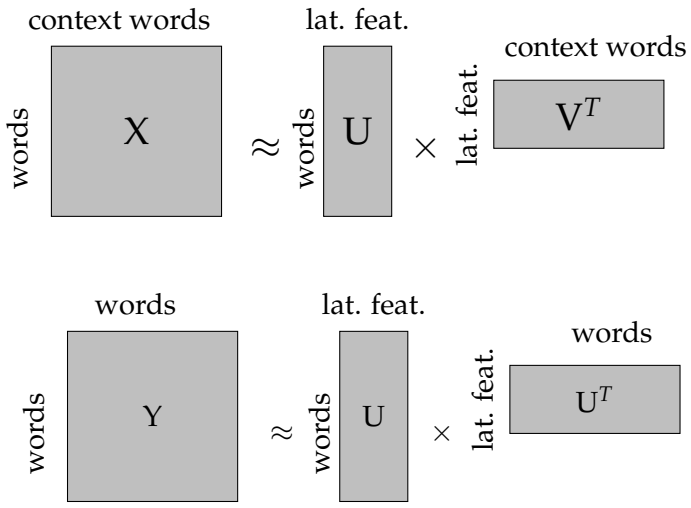

FIGURE 5.1: Visual overview of the matrix decomposition used for semantic similarity between pair of words on two matrices.

The first matrix is $X \in \mathbb{R}^{m1 \times n}$ where each row of X represents the distributional features of the first word of the pair. X has been approximated by the product of two smaller matrices U and V, where $U \in \mathbb{R}^{m1 \times k}$ is a matrix of the first words of the pair and latent features, and $V \in \mathbb{R}^{n \times k}$ is a matrix of distributional features as instance and the latent features. Here again, latent features size k has been chosen freely with $k << n$ for all matrices.

The second matrix is $Z \in \mathbb{R}^{m2 \times n}$ where each row of Z represents the distributional features of the second word of the pair. Z has been approximated by the product of R and V matrices, where $R \in \mathbb{R}^{m2 \times k}$ is a matrix of the second words of the pair and latent features, and $V \in \mathbb{R}^{n \times k}$ is a matrix of distributional features and the same latent features.

The third matrix is $Y \in \{0, 1\}^{m1 \times m2}$ which contains the semantic relation between each word of the pair. $Y_{i,j}$ has value 1 if the first word of the pair i is semantically related with the second word j, otherwise 0. Y has been approximated by the product of matrices U and R, where $U \in \mathbb{R}^{m1 \times k}$ is a matrix of first words of the pair and latent features, and $R \in \mathbb{R}^{m2 \times k}$ is a matrix of the second words of the pair and the same latent features.

More formally, X, Z and Y can be factorized as follows:

$$X \approx UV^T \tag{5.7}$$

$$Z \approx RV^T \tag{5.8}$$

$$Y \approx UR^T \tag{5.9}$$

The problem is now about how to minimize the objective function with respect to L2 loss function. The objective function is shown in Equation 5.11.

MRMF Method - Learning Algorithms and Predictions

Like semantic classification task, the semantic similarity task has also used block coordinate descent (CD) algorithm to optimize the objective function. As we have

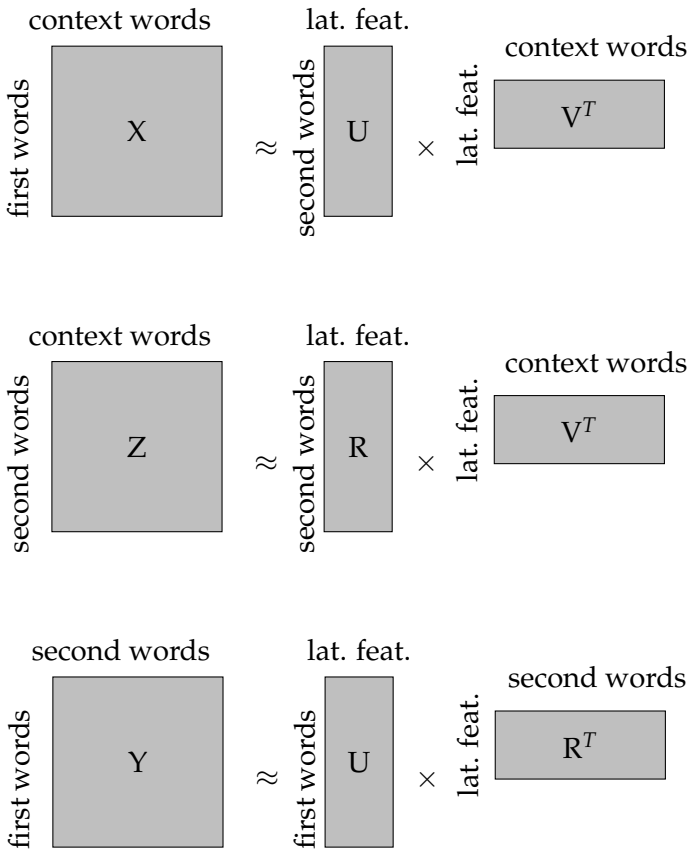

FIGURE 5.2: Visual overview of the matrix decomposition used for semantic similarity between pair of words on three matrices.

mentioned in the semantic classification chapter (Chapter 4), coordinate decent minimizes along coordinate directions to find the minimum of an objective function. This optimization algorithm has been the first proposed algorithm to minimize the objective function for both approaches. However, the first approach matrices design is not straightforward to calculate the derivation of the matrices by CD. Therefore, this algorithm has been used for the second approach only. This algorithm is described in Algorithm 3. The algorithm, first, initializes U, V and R matrices with random values. Then the minimization problem is solved for each matrix. This is repeated until convergence. This algorithm has been constructed to optimize the objective function which is shown in Equation 5.11.

For the first approach, stochastic gradient descent (SGD) algorithm has been used to optimize the objective function of the problem in Equation 5.10. SGD is a stochastic approximation of the gradient descent optimization, and iterative method for minimizing an objective function. The algorithm is described in Algorithm 4. In addition, the python script of MRMF on the SGD algorithm that we used in our study, is given in Appendix A. In this algorithm, $U_i \in \mathbb{R}^k$ denotes i-th row of U (which also can be interpreted as the latent features of the word i). Analogously, V_j and U_j represent the j-th row of V and U. Before solving the minimization problem, the U and V matrices are initialized with random values.

On both algorithms, after initializing the matrices (U,V and R) with random values, the minimization problem has been solved for each matrix. This is repeated

until convergence. Both algorithms CD and SGD have been optimizing the matrices with respect to L2 loss function as the functions are shown in Equation 5.10 and Equation 5.11 respectively. The L2 loss function is basically minimizing the sum of the square of the differences between the target value and the estimated values.

$$\begin{aligned}\underset{U,V}{\arg\min} \quad & \alpha_X \frac{1}{2}||X - UV^T||_F^2 + \alpha_Y \frac{1}{2}||Y - UU^T||_F^2 \\ & + \frac{\lambda_U}{2}||U||_F^2 + \frac{\lambda_V}{2}||V||_F^2\end{aligned} \tag{5.10}$$

$$\begin{aligned}\underset{U,V,R}{\arg\min} \quad & \alpha_X \frac{1}{2}||X - UV^T||_F^2 + \alpha_Z \frac{1}{2}||Z - RV^T||_F^2 + \alpha_Y \frac{1}{2}||Y - UU^T||_F^2 \\ & + \frac{\lambda_U}{2}||U||_F^2 + \frac{\lambda_V}{2}||V||_F^2 + \frac{\lambda_R}{2}||R||_F^2\end{aligned} \tag{5.11}$$

Algorithm 3 CD algorithm for L2-MRMF

procedure MRMF-COORDINATE DESCENT
input: X, Y, Z, k, *weight constants* $\alpha_x, \alpha_y, \alpha_z$, *regularization constants* λ_u, λ_v, λ_r
  $U \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $V \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $R \sim \mathcal{N}(0, \sigma\mathbf{I})$
  repeat
    $U \leftarrow \left(\alpha_x XV + \alpha_y YR^T\right)\left(\alpha_x V^T V + \alpha_y R^T R + \lambda_u \mathbf{I}\right)^{-1}$
    $V \leftarrow \left(\left(\alpha_x U^T U + \alpha_y R^T R + \lambda_v \mathbf{I}\right)^{-1}\left(\alpha_x U^T X + \alpha_x R^T Z\right)\right)^T$
    $R \leftarrow \left(\alpha_z ZV\left(\alpha_z V^T V + \lambda_r \mathbf{I}\right)^{-1} + \left(\left(\alpha_y U^T U + \lambda_u \mathbf{I}\right)^{-1}\alpha_y U^T Y\right)^T\right)$
  until convergence
  return U, V, R
end procedure

In both approaches, the matrices have been optimized by integrating the semantic relation of the pair of words information with the distributional information matrices. The matrices contain only the training pair of words. To predict the new pair of words semantic relation, Equation 5.12 has been used for the first approach and Equation 5.13 for the second approach.

$$Y_{ij}^{test} = \sum_l U_{il} U_{jl} \tag{5.12}$$

$$Y_{ij}^{test} = \sum_l U_{il} R_{jl} \tag{5.13}$$

Parameter Selection

A combination of learning rate, latent features and regularization parameters with a wide range of values have been tested to find the best parameter values setting. The

Algorithm 4 SGD algorithm for MRMF on two matrices to predict pairs

procedure MRMF-SGD
input: $X, Y, k, \alpha_X, \alpha_Y, \lambda_U, \lambda_V,$
  $U^{(0)} \sim \mathcal{N}(0, \sigma \mathbf{I})$
  $V^{(0)} \sim \mathcal{N}(0, \sigma \mathbf{I})$
  repeat
    $i \sim \mathcal{U}(1, m)$
    $j \sim \mathcal{U}(1, n)$
    $U_i \leftarrow U_i + \mu \left(2 * \alpha_X (X_{i,j} - U_i^T V_j) V_j + \lambda_U U_i\right)$
    $V_j \leftarrow V_j + \mu \left(2 * \alpha_X (X_{i,j} - U_i^T V_j) U_i + \lambda_V V_j\right)$
    $i \sim \mathcal{U}(1, m)$
    $j \sim \mathcal{U}(1, m)$
    $U_i \leftarrow U_i + \mu \left(2 * \alpha_y (Y_{i,j} - U_i^T U_j) U_j + \lambda_U U_i\right)$
  until convergence
  return U, V
end procedure

best parameter values minimize the objective functions and optimize the matrices. On the CD algorithm, learning rates α_x and α_z parameters value with a number in between the range $\frac{1}{\#X\_instances}$ and $1 \cdot 10^{-6}$, and in between the range $\frac{1}{\#Z\_instances}$ and $1 \cdot 10^{-6}$ have been considered respectively. The regularization constants λ_u, λ_v and λ_r have used the same range of value which is in between the range $1 \cdot 10^{-2}$ and $1 \cdot 10^{-10}$.

On the SGD algorithm, the learning rate μ has been learned on values [0.1, 0.01, 0.001, 0.0001,0.00001] with a decay values [0.9, 0.95, 0.98, 1.0]. The other learning rate α_x parameter value has been learned in between the range $\frac{1}{\#X\_instances}$ and $1 \cdot 10^{-6}$. The regularization constants λ_u and λ_v have used the same range of value which is in between the range $1 \cdot 10^{-2}$ and $1 \cdot 10^{-10}$.

For both algorithms, the learning rate of α_y has been set to 1; Because Y is the matrix that we are building the model to classify the pairs to the Y matrix semantic similarity relation information. Therefore, the full training information of the Y matrix is needed. For the latent features k, we have considered values in between the range 100 and 400. As Figure 5.3 shows, parameter value k = 300 has been giving good result on SGD algorithm. The k parameter has been performed better with the μ value (0.01) , decay value (0.95) and α_x value (0.01). On CD algorithm, the k parameter value has been performing better when the value is 200. The performance has been better when α_x and α_z parameter values are 1e-06 and 0.0001 respectively. For both algorithms, the lambda values have been small which is close to zero to get the highest performance.

5.4 Result

The results of the HsH-Supervised and the two baselines have shown in Table 5.3 on F1-score and accuracy percentage. The evaluation script has already given for the shared task. Therefore the experiments have been evaluated using the given evaluation script. The script evaluates the systems and gives their F1-Score result as an output. The systems accuracy percentage which is given in the result table collected from SVM.

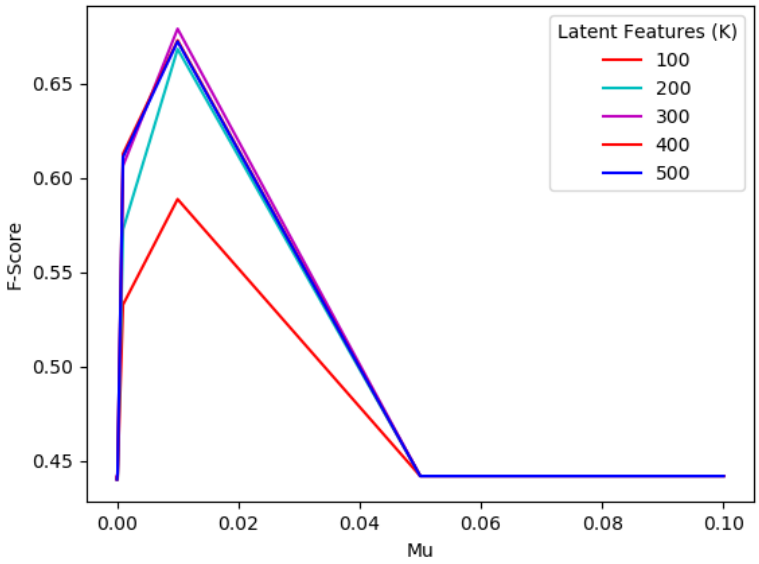

FIGURE 5.3: Accuracy of MRMF on SGD Algorithm with different k and μ parameters value

As the result Table 5.3 shows, for the first sub-task, the HsH-Supervised could not give better result comparing the simple cosine similarity baseline. However, it has outperformed the *addition* system. In addition, it is clearly better than the naive baseline that considers each pair to be related. However, if we see the accuracy result of the HsH Supervised system and cosine, they have only a 0.02 difference. Nonetheless, the cosine outperforms the HsH supervised system. For the second sub-task, the F1-score of the HsH-Supervised is very low, but still far above the naive baseline. Remarkably, the precision is quite high, and half of the pairs have found for one of the four semantic relations, indeed have this relation. High precision relates to the low false positive rate.

Task	Method	Precision	Recall	F-score	Accuracy
Task 1	All True (Majority)	0.282	1.000	0.440	0.282
	Cosine	0.590	0.713	0.646	0.780
	Addition	0,362	0.094	0,149	0,698
	HsH-Supervised	0.577	0.593	0.585	0,760
Task 2	All hypernym (Majority)	0,0897	0,318	0,140	0,090
	HsH-Supervised	0.506	0.154	0.229	0,753

TABLE 5.3: Performance of the HsH-Supervised and two baselines for both tasks

The MRMF method experiments have also been evaluated using the given Evaluation script to compare the results directly with the published systems result. In addition, its accuracy percentage has also been calculated. The MRMF method results have shown in Table 5.4. Since the MRMF method has been studied after the publication of the HsH-Supervised result, the MRMF method results can be compared with the rest of the published systems results as well. This method has been applied only to the first given shared sub-task. This is because even if the result on the first sub-task is better than the HsH-Supervised and Cosine method (which is

one of the baselines), it did not take the first rank (place) of the competition. There are still four methods (GHHH, Mach5, LexNET, ROOT18 [2]) on top of the MRMF method result just like the HsH-Supervised result. However, the MRMF method has been found to outperform the HsH-Supervised and the baseline (Cosine similarity measure). From the two approaches of MRMF method, the second approach which integrates the three information and builds a model on SGD has outperformed the first approach (CD) as the result table shows. The three integrated information in the second approach is the first word of the pair, the second word of the pair and the semantic relation of the pair of the words.

System	**F-score**
MRMF (SGD)	0,680
MRMF (CD)	0,605
Cosine	0,64
Published Systems and Results	
GHHH	0,790
Mach5	0.778
LexNET	0,765
ROOT18	0,731
LOPE	0,713
HsH-Supervised	0.585
CGSRC	0,431

TABLE 5.4: Performance of the MRMF method on the first given shared subtask

The top five systems (GHHH, Mach5, LexNET, ROOT18, LOPE) which have outperformed the HsH-Supervised and the MRMF method have explained briefly as follows. However, the explanation focuses more on the systems first sub-task method to compare our MRMF method as well which has implemented only in the first sub-task. Many of the methods have used two different methods for both sub-tasks separately.

The GHHH system (Attia et al., 2016) is the top-ranked method for the first sub-task and ranked second on the second sub-task. This system uses the publicly available pre-trained English word vectors representations. The system simply uses Simple Logistic which is linear regression algorithm to train the model. First, each word has represented by different word embeddings. Then, the cosine distance has computed between the pair of words vector representation. The cosine distance result of the pairs on each word embedding representation have aggregated and represent the pairs. Finally, the model has been trained on the aggregated cosine distance results. The word embeddings are Google News [3] and Wikipedia + Gigaword 5[4]. This system is used for the first task but for the second task, Convolutional Neural Network (CNN) architecture is used. GHHH has used two different methods (linear regression and CNN) for both sub-tasks.

The Mach5 system (Evert, 2016) is the second-ranked system for the first sub-task and third-ranked for the second sub-task. This system uses the same approach

[2] https://sites.google.com/site/cogalex2016/home/shared-task/results
[3] https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/
[4] http://nlp.stanford.edu/data/glove.840B.300d.zip

as our system to represent each word by a vector which is the co-occurrence distributional semantics representation. However, the system has used different parameters tuning and different corpus. They used ENCOW English web corpus (Schäfer and Bildhauer, 2012) which contains 9.5 billion tokens.The word has represented on both a dependency-filtered (*DepFilt*) and a dependency-structured (*DepStruct*) that is compiled from syntactic dependencies obtained with the C&C parser (Curran, Clark, and Bos, 2007). The 120,000 most frequent lemmas were extracted as features for the (*DepFilt*). The 300,000 most frequent relation-lemma combinations were extracted as features for the *DepStruct*. The system follows the Lapesa, Evert, and Schulte im Walde (2014) framework by constructing the distributional features representation. The basic parameters of Lapesa, Evert, and Schulte im Walde (2014) are: using simple log-likelihood (simple-ll) as an association measure for feature weighting and an additional log transformation is applied to the simple-ll scores. In addition, medium-frequency features have used. Then, words on DepFilt have represented with 50,000 features (except for the 150 most frequent lemmas), the words on DepStruct have represented with 100,000 features (except for 400 most frequent ones). Then, the SVD dimension reduction method has applied on the *DepFilt* and *DepStruct* vectors. Finally, each word vector representation is reduced to the first 600 SVD dimensions without power scaling (P = 1). They experimented different methods for the system. For the first sub-task, the system uses distance information (an angular distance which is similar to cosine distance) from the DepFilt which the optimal cutoff threshold is determined to maximize F1 on the training data. For the second sub-task, features from DepFilt and DepStruct are concatenated for a total of 1,200 feature dimensions. Then, a linear SVM classifier has applied to feature vectors which contain the contribution $(a_i - b_i)^2$ of each latent dimension i to the Euclidean distance between the pre-normalized vectors first words x and y of a word pair.

The LexNet system (Shwartz and Dagan, 2016) is the third ranked system for the first sub-task and fourth ranked for the second sub-task. This system is an integrated path-based and distributional method for semantic relation classification. It represents pairs as a feature vector which is consisting of distributional and path-based features. The mathematical representation of the systerm is $\vec{V} = [\vec{A}, \vec{V}_{paths(x,y)}, \vec{B}]$, where $\vec{A}$ and $\vec{B}$ are the first and second words of the pair embedding which provides distributional representation, and $\vec{V}_{paths(x,y)}$ is the average embedding vector of all the dependency paths that connect words of the pair A and B in the corpus. The word embedding which is used in this system, is 50-dimensional pre-trained GloVe word embedding (Pennington, Socher, and Manning, 2014) which is trained on Wikipedia and Gigaword 5 (6B tokens). The dependency paths are embedded using a LSTM (Long Short Term Memory) (Hochreiter and Schmidhuber, 1997). This vector is used as an input for a neural network that outputs the class distribution $\vec{c}$, and then the pair is classified to the relation with the highest score r. Notations: MLP is Multi Layer perceptron, W_1 and b_1 are the network parameters. The system performed better when the hidden layer model is zero.

$$r = argmax_i c[i] \tag{5.14}$$

$$\vec{c} = softmax(MLP(\vec{V}_{xy})) \tag{5.15}$$

$$MLP(\vec{V}) = W_1 \cdot \vec{V} + b_1 \tag{5.16}$$

For the first task, the system integrates the cosine distance between the pair of words as well with LexNet score. To compute the cosine similarity, the word vectors have

chosen among several available pre-trained embedding [5]. Then, they score the pair by a combination of LexNet's score for each related class and the cosine distance as shown below: where, w_C and w_L are the weights assigned to cosine distance and LexNet's scores respectively.

$$Rel(x,y) = w_C \cdot cos(\vec{A},\vec{B}) + w_L \cdot \vec{c}[i] \tag{5.17}$$

The pairs are classified as related when $Rel(x,y) \geq t$; otherwise, not related. The weights and a threshold t have tuned using the validation set. For the second task, the system uses just only $LexNet$.

The ROOT18 system (Chersoni, Rambelli, and Santus, 2016) is the fourth ranked system for both sub-tasks. This system is simply a Random Forest classifier and it is based on 18 features. These features are:

- Frequency: For each pair, three features have computed which are the frequency of each word (Freq1,2) and their difference ($DiffFreq$).
- Co-occurrence: co-occurrence frequency (Cooc) between the two words in each pair.
- Entropy: For each pair, three features have been computed which are the entropy of each word in the pair ($Entr1,2$), plus the difference between the entropies ($DiffEntr$).
- Cosine similarity: compute the similarity distance between words of each pair
- LinSimilartiy: It is computed as the ratio of shared context between x and y to the contexts of each word (Hovy and Lin, 1998):

$$Lin(\vec{A},\vec{B}) = \frac{\sum_{c\in\vec{A}\cap\vec{B}}[\vec{A}|c] + \vec{B}[c]}{\sum_{c\in\vec{A}}\vec{A}[c] + \sum_{c\in\vec{B}}\vec{B}[c]} \tag{5.18}$$

- Directional similarity: four directional similarity measures that were proposed to detect hypernyms, such as $WeedsPrec$, $cosWeeds$, $ClarkeDe$, and $invCL$. These methods have explained in detail in Lenci and Benotto (2012).
- APSyn: It computes a weighted intersection of the top N context of the target words. Where $F1$ and $F2$ are the top N features of words x and y.

$$APSyn(x,y) = \sum_{f\in N(F1)\cap N(F2)} \frac{1}{(rank_1(f)) + rank_2(f)/2} \tag{5.19}$$

- APAnt: It is the opposite of APSyn. This method explained in their paper Santus et al. (2014).
- Same POS: It contains a boolean value: 1 if the most frequent POS (part of speech) of the words in the pair are the same, 0 otherwise.

[5] 4word2vec (300 dimensions, SGNS, trained on GoogleNews, 100B tokens) (Mikolov et al., 2013b), GloVe (50-300 dimensions, trained on Wikipedia and Gigaword 5, 6B tokens) (Pennington et al., 2014), and dependency-based embeddings (300 dimensions, trained on Wikipedia, 3B tokens) (Levy and Goldberg, 2014)

The LOPE (Luce, Yu, and HSIEH, 2016) is the fifth ranked system for the first sub-task and last ranked for the second sub-task. For the first sub-task, they have proposed to use word vectors and calculate similarity of the pairs. The words have represented by the pre-trained Google News vectors (Mikolov et al., 2013a). Then, the similarity distance has calculated between the words of the pair vector representation. For the second sub-task, they have used a naive approach, by assigning word pairs semantic relations based on their parts of speech. They have followed the nearest neighbor position indexing by assuming the order *synonymy, antonymy, hypernymy, meronymy and random*.

In general, each system has used different methods for each sub-task. In addition, except Mach5, the rest systems have used the pre-trained word embeddings. Each system came up with different interesting approaches. But each of them has to be investigated to find the reason behind getting the top rank comparing our systems HsH supervised and MRMF. Two main points have investigated. 1. The corpus size that has used to represent the words by a vector (word embedding), 2. The method which has used to train the model. Table 5.5 shows the summery of the systems to make the conclusion on the above two points. As the table shows, all systems used very big corpus except ROOT18 which has used 2B corpus size like our systems. However, the lowest ranked system on the first sub-task used 100B corpus size. Therefore, it's not only the corpus size which gave them top rank. Constructing the word vector's representation on a big corpus and the methods to build the classifier have their own impact together for the better result. Specially, as Bullinaria and Levy (2012) stated in their study, constructing vector representation on a big size corpus gives high quality vector representation.

Published Systems	**Corpus Size** (#tokens)	**Method**
GHHH	106B	SVM (***input:*** cosine distance from two word embeddings)
Mach5	9.5B	Euclidean Distance
LexNET	100B	Multi Layer Perceptron + Cosine
ROOT18	2B	Random Forest (***input:*** 18 similarity measures)
LOPE	100B	Cosine Distance
HsH-Supervised	2B	SVM
CGSRC	100B	CNN

TABLE 5.5: Performance of the MRMF method on the shared task

5.5 Conclusion

The distributional vector for the pair of words can be generated by multiplication (or using other mathematical operations). Pairwise multiplication is often in use to represent pair of words by distributional vectors. Then, the pairs vector can be used to train a supervised model which learns whether or not the words in the pair are semantically-related. This method has been applied to the CogALex shared task. The standard SVM has been used which optimizes the overall accuracy, while the given official evaluation is the F1-Score on a small class. In fact, the accuracy is quite high and the difference in accuracy between the simple cosine based method and the supervised method is very small. Finally, we have the impression that the method is

successful in recognizing a loose semantic relatedness, but is not able to distinguish between very closely related words (like synonyms) and more loosely related words. In our previous semantic similarity study, we have studied the relatedness of words in a thesaurus. Here the supervised method also performs well on a pair of words that are related to each other by some thesaurus relations via at most one intermediate concept. The performance is not good on pairs built from alternative labels for the same concept. Here we have a similar situation, in which we only want to find words with a specific and precisely defined semantic relation, while other words that have alternative or looser semantic relations which are classified as unrelated. Thus, it seems that the findings of the present experiment are in-line with previous results for the same approach.

In conclusion, the HsH-supervised system has ranked fifth on the computation from the total of six participants for the first task. The method which has been proposed on the second phase (MRMF) has performed better than the HsH-Supervised method and the baselines, but not above the top systems (GHHH, Mach5, LexNet, ROOT18). Since the challenge was corpus-based challenge, MRMF method has not been integrated with a different source of information such as WordNet to improve the performance of the model. In general, MRMF can be used for semantic similarity task since it has shown better performance when compared to HsH-supervised. However, further studies are nonetheless required on different types of pairs which are not challenging like Co-gALex pairs. Furthermore, additional sources of information can be integrated with the distributional information using MRMF for better performance of the model, and also different matrices structure approach can be investigated to integrate the relations between the data.

Chapter 6

Phrasal Semantics

6.1 Introduction

In the study of semantics, phrasal semantics is one of the main areas beside lexical semantics. Phrasal semantics concerns the meaning of syntactic units larger than the word, while lexical semantics is concerned with the meanings of words and the meaning of relationships among words. Phrasal semantics is the study of the principles which determine the construction of the meaning of phrases and a sentence meaning out of compositional combinations of individual words (Firm, 2010).

This chapter discusses the phrasal semantics study which is a task beyond lexical terms. The aim of this study is to investigate the multi-relational matrix factorization method performance on phrasal semantics. The multi-relational matrix factorization (MRMF) method has been showing good performance in our previous studies on the lexical semantics tasks such as semantic classification and semantic similarity. Therefore, this method study has been extended to a phrasal semantics task.

The idea behind the phrasal semantics task is computing the semantic similarity of words and compositional phrases of minimal length. Thus, the main idea of this study is to build a model using MRMF method which computes the semantic similarity between words and compositional phrases. The phrasal words are a maximum of three. Basically, the first word is the semantic relation of two compositional words.

For this study, the SemEval-2013 Task 5[1] is selected. This task is called *Evaluating Phrasal Semantics*. The SemEval-2013 Task 5 has been used as a literary work to study the performance of MRMF method on phrasal semantics. SemEval-2013 Task 5 has two sub-tasks. From the two sub-tasks, phrasal semantics task is one of the sub-task that has been used to study the performance of MRMF method. This sub-task is written as *SemEval-2013 Task 5a*.

SemEval-2013 Task 5 is a shared task. For this shared task, around ten participants have proposed different systems. The participants' studies and their systems rank are summarized under Korkontzelos et al. (2013) paper. The highest ranked method is called *HsH*. This system has used in our study as a baseline. It has explained in detail in the following sections with the MRMF method.

The rest of the paper is organized as follows. Section 6.2 discusses the SemEval-2013 Task 5. Section 6.3 explains the given data. The methodology has discussed in Section 6.4 which contains the highest ranked method (HsH) and MRMF method explanation. Result and Discussion with conclusion have been explained in Section 6.5 and Section 6.6 respectively.

[1] https://www.cs.york.ac.uk/semeval-2013/task5.html

6.2 Evaluating Phrasal Words

Evaluating Phrasal Semantics is the SemEval-2013 Task 5 task. Evaluating Phrasal Semantics has two sub-tasks. These are: 1. computing the semantic similarity of words and compositional phrases which have minimal length such as two; it is referred to as *SemEval-2013 Task 5a*. 2. evaluating the compositionality of phrases in context which is deciding the compositionality of phrases in a given context; it is referred to as *SemEval-2013 Task 5b*.

The scientific benefit of this shared task and the aim of studying the two subtasks have described in two parts. The two parts have explained in their task description[2] as follow: *1. It provides an opportunity to draw together approaches to numerous related problems under a common set of evaluations. It is intended that after the competition, the evaluation setting and the datasets will comprise an ongoing benchmark for the evaluation of these phrasal models. 2. They anticipated that these tasks will stimulate increased interest the general issue of phrasal semantics by bridging the gap between established lexical semantics and full-blown linguistic inference. This could provoke certain established tasks such as lexical entailment and paraphrase identification, and ultimately lead to improvements in a wide range of applications in Natural Language Processing, e.g. document retrieval, clustering and classification, question answering, query expansion, synonym extraction, relation extraction, automatic translation, or textual advertisement matching in search engines, all of which depend on phrasal semantics.*

This study is focused on the first sub-task, which is SemEval-2013 Task 5a. The original aim of this subtask is to evaluate how well systems can judge the semantic similarity of a word and a short sequence of words (like two words) (Korkontzelos et al., 2013). For example: *(dreamland, imaginary world)*; the meaning of the sequence *imaginary world* as a whole is semantically-related to the meaning of the word *dreamland*. When the meaning of the word is semantically different to the meaning of the sequence, it is unrelated phrasal semantic words; For example: (*facebook*, *fireplace accessory*). In the example, the sequence *fireplace accessory* as a whole is not semantically related to the word *facebook*. In general, the participating systems are asked to estimate the semantic similarity of a word and a short sequence of two words. In our study, this subtask investigates the performance of the MRMF method on extracting semantic relation in phrasal words.

From the list of methods which have proposed to solve the Evaluating Phrasal Semantics shared task, the *HsH method* is one of the method which took the highest rank comparing the rest of the participants' methods. This method result is taken as the baseline in this study to evaluate the performance of MRMF method, as mentioned briefly in the introduction.

The HsH method, in order to get the highest-ranked result, built a model by using distributional similarities of the phrasal words. Before the model is trained, various distributional similarity measures have used to compute the phrasal words' similarity. Then the distributional similarities results have been concatenated as features designed to represent the phrasal words by a vector. Basically, the distributional similarities are used to represent the semantic relation of the single word with the two words sequence. Afterward, the model is trained on a supervised learning algorithm by taking the phrasal words vector representation as an input. Finally, the model is used to classify the single word with the two words phrase semantic relation as related or not related. The method is explained in detail in section 6.4.2.

[2] https://www.cs.york.ac.uk/semeval-2013/task5.html

6.3 Data

For the purposes of evaluating the Phrasal Semantics shared task, training and test datasets have been provided. Thus, these datasets have been used directly in our study. The datasets which have given for the first subtask targets on three different languages - English, German and Italian. As Korkontzelos et al. (2013) explained that the word-sequence pairs of the datasets prepared from the English, German and Italian Wiktionary by downloading all Wiktionary entries. Wiktionary is a multilingual, web-based project designed to create a free content dictionary of all words in all languages. Then, part-of-speech has been tagged using the Genia tagger (Tsuruoka et al., 2005). Afterward, to extract the word-sequence pairs, noun phrases have been considered. Specifically, sequences that consist of adjectives or nouns and end with a noun have been considered.

Even if the datasets are available in three languages, in our study, we consider the English language datasets. For the first subtask, datasets which contain positive and negative word-sequence pairs, have provided. 60% of the data is annotated as positive/negative for training and, 40% unannotated for testing set. The total number of the training set is 11, 722 annotated examples of training pairs, and 3,906 unannotated examples of a test set. The distribution of the negative and positive examples of the training set is equal; 5,861 positive and 5,861 negative examples are given. The test set is constructed by a random selection of an equal number of 1953 positive and 1953 negative examples from two labeled files. Then the randomly selected word-sequence pairs put together in a single file. For the random selection, the two labeled files contain 3907 positive and 3907 negative examples separately. The sample data is shown in Table 6.1.

Word	Sequence pair	Class
time	particular moment	+
information	abstract datum	+
service	religious rite	+
day	rotational period	+
area	geographic region	+
area	open space	+
group	functional entity	+
child	datum item	+
world	collective existence	+
business	particular situation	+
time	routine inspection	-
information	arrogant courage	-
service	particular style	-
day	decorative strip	-
area	lodge officer	-
area	tin foil	-
group	male human	-
child	positive result	-
world	inferior composition	-
business	keyboard key	-

TABLE 6.1: The SemEval-2013 Task 5a sample data

6.4 Methodology

This section explains the HsH system which took the highest score in SemEval-2013 shared Task 5, and MRMF method. The HsH system proposed by Wartena (2013). Wartena (2013) followed a supervised learning approach which uses distributional similarities information as features in order to predict the semantic relatedness of a single word with two-word phrases. The second method, MRMF, has been selected to evaluate its performance for a task beyond lexical words. The MRMF method has been explained in the previous Chapters 4 and 5 on the studies of semantic classification and semantic similarity respectively. Even if MRMF follows the same terminology and idea, some details of the method are different in different tasks. MRMF can integrate different sources of information easily and effectively and guarantee the high quality of models for the given tasks.

Before the methods detail explanation, we can see the following examples to understand the phrase semantic task, clearly, that has considered in this study clearly.

Example-1: Let's take two semantic composition words *Dog* and *House*, and their semantic related word is *Kennel*.

Dog and *House* $\rightarrow$ *Kennel*

Example-2: Let's take two semantic composition words *Royal* and *House* , and their semantic related word is *Palace*.

Royal and *House* $\rightarrow$ *Palace*

The following subsections explain the HsH and MRMF methods in detail with the words' representation. Subsection 6.4.1 explains the two methods which have used to represent the experiment dataset words. Subsection 6.4.2 explains the HsH method which is the state of the art system. Finally, subsection 6.4.3 explains the MRMF method in detail with its matrices structure and algorithms.

6.4.1 Word representation

The experiments have been started by representing each word of the dataset by a context vector. Wartena (2013) represented each word by following a random indexing technique which has introduced by Karlgren and Sahlgren (2001) and Sahlgren (2005). The main idea of Random indexing technique is to accumulate context vectors based on the co-occurrence of words in contexts (Sahlgren, 2005). If this technique is used for word representation, separate dimension reduction phase does not need (Sahlgren, 2005). Random indexing techniques have two steps which have been explained in (Sahlgren, 2005) to construct a context vector. These are: 1. Each word is assigned by d dimension index vector which contains unique or randomly distributed small numbers +1s, 0s and -1s.; 2. A context vector is produced by scanning through the text and each time a word occurs in a context (e.g. corpus), that context's d-dimensional index vector is added to the context vector for the word in question. The context vector was constructed using the complete Wacky corpus (UkWaC). When Wartena (2013) represents each word by context vector, all open-class words (i.e. Noun, Verb, Adjective, Adverb, etc) used as a context word, but not Auxiliary, Pronoun, etc.). Finally, each word is represented by a 10 000 dimensions vector.

In our study, each word has been represented by context vector which is constructed by following the word representation explanation in Chapter 2 Section 2.5.1. Then each word is represented by the 17 400 dimensions vector.

6.4.2 HsH Method

HsH method trains a model on a supervised learning algorithm by taking the phrasal words vectors representation as an input. The phrasal words vector representation is constructed by concatenating results from various distributional similarity measures which have been applied to the phrase words. The distributional similarity measures compute the similarity distance between the single word and the two semantic compositional words (sequence pair).

To explain the distributional similarity measures, let us bring in some variables definitions which have used in Wartena (2013). The two semantic composition words (the sequence pair) are referred to as $d1$ and $d2$. The semantically related word of the two semantic composition words is referred to as w. The two semantic composition words together are referred to as d. This means $d = (d1, d2)$. The d vector $\vec{v}_d$ has been computed in three different ways: 1. by computing the context vector directly from the corpus even if it has very few occurrences; 2. using the simple mathematical operation addition: $\vec{v}_d^{add} = \vec{v}_{d1} + \vec{v}_{d2}$; 3. using multiplication: $\vec{v}_d^{mul} = \vec{v}_{d1} \cdot \vec{v}_{d2}$.

The main similarity distance measures, which have used in the HsH system are *Jensen-Shannon Divergence (JSD)*, *Cosine (cossim)* and *co-occurrence ratio (co-occ-ratio)*. These three similarity measures have applied on the context vectors of each word of the phrases in different ways. Then, eleven features have constructed from the three measures to represent the phrases. JSD has applied on two different forms which are normalized and not normalized (the normal JSD). The normal JSD formula is shown below in Equation 6.1:

$$JSD(p,q) = \frac{1}{2}D(p||\frac{1}{2}p + \frac{1}{2}q) + \frac{1}{2}D(q||\frac{1}{2}p + \frac{1}{2}q) \tag{6.1}$$

Where $D(p||q) = \sum p(i)\frac{logp(i)}{logq(i)}_i$ which is the KullbackLeibler divergence. The Wartena (2013) new JSD measure is normalized JSD. The normalized JSD , first, models the dependency between the JSD and the number of occurrences of the involved words. Then we take the difference between the JSD of the co-occurrence vectors of two words and the JSD expected (JSD^{exp}) on the base of the frequency of these words as a similarity measure. The normalized JSD is shown in Equation 6.2 with the JSD^{exp} equation. Given two words $w1$ and $w2$ the JSD of their context vectors can be modeled as a function of the minimum of the number of occurrences of $w1$ and $w2$. Then, JSD^{exp} defined for the context vectors c_1 and c_2 for the words w_1 and w_2 respectively.

$$\begin{aligned} JSD^{exp}(c_1, c_2) &= a + \frac{1}{\hat{n} + c} \\ JSD^{norm}(p,q) &= JSD(p,q) - JSD^{exp} \end{aligned} \tag{6.2}$$

In the equation, where $\hat{n} = min(n(w1), n(w2))$ with $n(w)$which is the number of occurrences of w in the corpus and with a,b and c constants that are estimated for each set of word pairs (Wartena, 2013). $a = 0.15, b = 0.3$ and $c = 0.5$ are values for the pair from the training and test set Wartena (2013).

The cosine similarity measure is applied between $\vec{v_w}$ and $\vec{v}_d^{add}$. In addition, it has applied between $\vec{v_w}$ and $\vec{v}_d^{mul}$. The last similarity measure *co-occ-ratio* use the direct (first-order) co-occurrence between w and d by computing the ratio between the probability with which w and d expect to co-occur in one sentence if they would

be independent, and the real probability of co-occurrence found in the corpus. The equation is shown as follow:

$$cooccratio(w,d) = \frac{p(w,d)}{p(w) \cdot p(d)} \tag{6.3}$$

In general, Table 6.2 shows the two similarity measures (JSD and Cosine) with the context vectors. In the table, the context vectors which have used the measures have ticked. Except, the cooccratio method, the remaining ten methods which are constructed by the cossim and JSD similarity measures, are shown in Table 6.2. This table is taken from Wartena (2013) paper directly. The detail explanation of these features can be found in Wartena (2013) paper.

Table 6.3 shows the eleven features which have constructed from the three similarity measures. The eleven similarity measure results are considered as features, and integrated in order to represent the phrasal words.

	V_d	V_{d1}	V_{d2}	V_d^{add}	V_d^{mul}
JSD	✓	✓	✓	✓	
JSD^{norm}	✓	✓	✓	✓	
cossim				✓	✓

TABLE 6.2: Similarity measures used to compute the similarity of a context vector of some word to various context vectors for a phrase d. (Wartena, 2013)

#	Feature
1	$\text{jsd}(\vec{v_w}, \vec{v_d})$
2	$\text{jsd}^{norm}(\vec{v_w}, \vec{v_d})$
3	$\text{jsd}(\vec{v_w}, \vec{v_{d1}})$
4	$\text{jsd}^{norm}(\vec{v_w}, \vec{v_{d1}})$
5	$\text{jsd}(\vec{v_w}, \vec{v_{d2}})$
6	$\text{jsd}^{norm}(\vec{v_w}, \vec{v_{d2}})$
7	$\text{jsd}(\vec{v_w}, \vec{v_d^{add}})$
8	$\text{jsd}^{norm}(\vec{v_w}, \vec{v_d}^{add})$
9	$\text{cossim}(\vec{v_w}, \vec{v_d}^{add})$
10	$\text{cossim}(\vec{v_w}, \vec{v_d}^{multi})$
11	$\text{co-occ-ratio}(\vec{v_w}, \vec{v_d})$

TABLE 6.3: The eleven features from the three similarity distance. (Wartena, 2013)

Finally, the eleven features have been concatenated in order to represent phrases. Then, a classifier has been trained on the phrases representation. The classifier classifies the phrases if the single word and sequence pair semantically related or not. To build the classifier, SVM algorithm is used. The algorithm took the eleven features as an input. The performance of this system (HsH) has used as a baseline, in our study to evaluate the MRMF method.

6.4.3 MRMF Method

MRMF method has been used for semantic similarity and semantic classification tasks in our previous studies as explained in the previous chapters (Chapter 4 and Chapter 5 respectively). To use this method for the phrasal semantics task, the same procedure has been followed.

MRMF has easy and effective nature of integrating different sources of information and/or different relations of entities by a mathematical computation. By taking this into consideration, an approach has been designed regarding the matrices type relations and structure. The approach contains three matrices which contain distributional information and one tensor which contains a semantic relation of the phrasal words. In this approach, the words of the phrases have split. As it is known, the phrase contains three words. Thus, three words have found after the split. The first word is the semantically related word of the two semantic composition words. The second word is the first word from the two semantic composition words. The third word is the second word from the two semantic composition words. Then, three matrices have prepared to represent the first, the second and the third word. The visual overview of this approach is shown in Figure 6.1 with the matrices and tensor decomposition .

In the approach, each of the three matrices contains the context vector representation of each word. This means that the three matrices contain distributional information of the words. The fourth matrix is a tensor. A Tensor can be represented as an organized multidimensional array of numerical values. The dimension of the tensor is called rank or order or degree. The order of a tensor is the dimension of the array that needs to be represented. For example, A 2-dimensional array is a matrix, therefore, it is a 2nd-order tensor; A 1-dimensional array represents a vector, therefore it is a 1st-order tensor; A 0-dimensional array is scalars which are single numbers, therefore, it is 0-order tensors. In our study, a tensor represents a cube which is an array with three dimensions, or it is called 3rd-order tensor. The tensor contains the semantic relation information of the phrasal words. Then, MRMF has been used to integrate the matrices with distributional information and the tensor with semantic relation information.

Let's assume the following problem, first, to see the method approach in detail. We have:

- $m1$: the semantically related word of the two semantic composition words.
- $m2$: the first word from the two semantic composition words.
- $m3$: the second word from the two semantic composition words.
- n features for each word (the distributional information i.e. positive pointwise mutual information (PPMI) values based on the co-occurrence data).

The features' representation of the three matrices, and also the tensor have been defined as follows:

- $Z^1 \in \mathbb{R}^{m1 \times n}$ where each row of Z^1 represents the feature vector of a word ($m1$);
- $Z^2 \in \mathbb{R}^{m2 \times n}$ where each row of Z^2 represents the feature vector of a word ($m2$);
- $Z^3 \in \mathbb{R}^{m3 \times n}$ where each row of Z^3 represents the feature vector of a word ($m3$).
- $Y_{i,j,k}$ has value 1 if the word i is semantically related with the words j and k, otherwise 0.

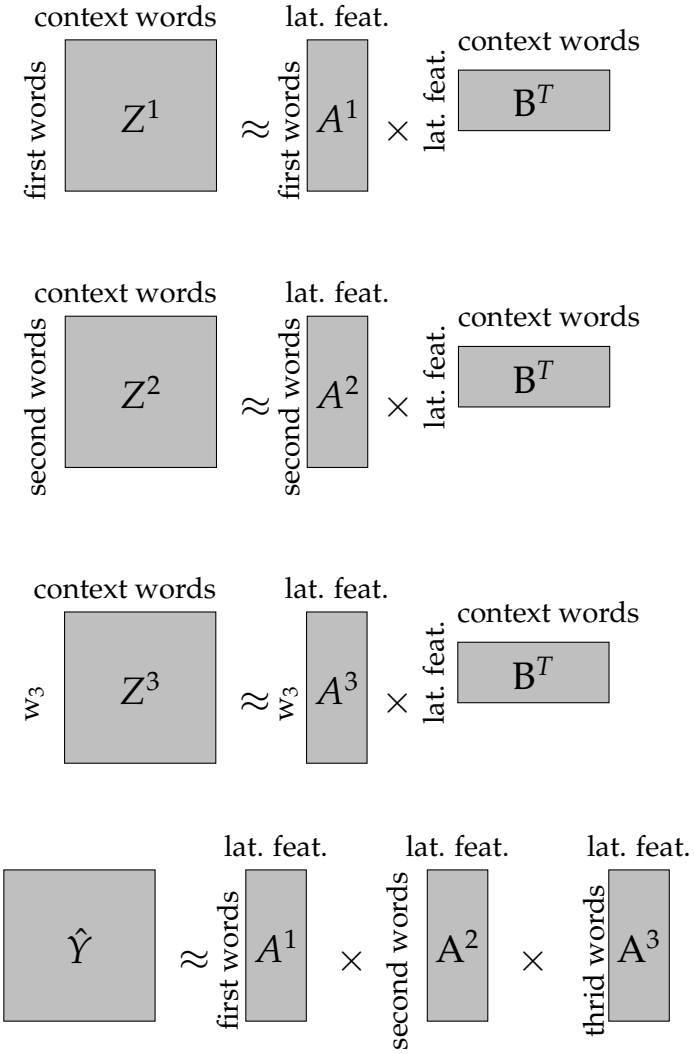

FIGURE 6.1: Visual overview of the matrices and tensor decomposition of the semantic similarity between a single word and two semantic composition words on four matrices.

The idea of matrix factorization in this approach is that Z^1, Z^2, Z^3 can be approximated by the product of two smaller matrices (i.e. A^1 and B ,A^2 and B, A^3 and B respectively). A^1,A^2,A^3 are matrices of the first, the second and third of phrasal words respectively and latent features. B is a matrix of context features(context words) and the same latent features. The number of latent features size k can be chosen freely with $k << n$. The Y tensor can be approximated by the product of three matrices (A^1,A^2,A^3).

The idea behind the MRMF is that both decomposition of the (Z^1, Z^2, Z^3) and Y use the same factor matrices A^1,A^2,A^3 which are words by latent features. Thus, the latent features form a link between the context features and the semantic relation information of the words.

More formally, Z^1, Z^2, Z^3 and $Y_{i,j,k}$ can be factorized as follows:

$$\begin{aligned} Z^1 &\approx A^1 B^T \\ Z^2 &\approx A^2 B^T \\ Z^3 &\approx A^3 B^T \\ \hat{Y}_{i,j,k} &= \sum_l A^1_{i,l} * A^2_{j,l} * A^3_{k,l} \end{aligned} \tag{6.4}$$

MRMF Method - Learning Algorithms and Predictions

Two optimization algorithms have been proposed to build a model that predicts the semantic relatedness of phrases (the word and the sequence pair) using MRMF method. The first algorithm is Alternating Least Squares (ALS), and the second algorithm is stochastic gradient descent (SGD). Each optimization algorithm has explained below in detail.

ALS is an algorithm to find an appropriate factorization of a large matrix, when the objective function is given in the least square sense. The mathematical definition of least square is a statistical method that is used to determine a line of best fit by minimizing the sum of squares created by a mathematical function. The ALS algorithm is shown in Algorithm 5. As the algorithm shows that it has new variables such as F_1, F_2, F_3, Y^1, Y^2 and Y^3. These variables definition has been shown below:

- $\hat{Y} = A^r F_r^T$ (F_r is the matricization of the $A^{r'}, r' \neq r$)

  Y^r can be denoted as the corresponding matricization of Y

  For each r = 1,2,3; Y will be rewritten, then it becomes the matrix case:

  $0 = \partial_{A^1} f(...) = -2(Z^1 - A^r B^T)B + 2\lambda A^r - 2(Y - A^r F_r^T)F_r$

- $A^r = (B^T B + F_r^T F_r + \lambda I)^{-1}(Z^r B + Y^r F_r)$

  The matricizations are defined as follows:

  - $r = 1 : (Y^1)_{n_1, n_2 N_3 + n_3} := Y_{n_1,n_2,n_3}$
    $(F^1)_{n_2 N_3 + n_3, k} := A^2_{n_2,k} A^3_{n_3,k}$
  - $r = 2 : (Y^2)_{n_2, n_1 N_3 + n_3} := Y_{n_1,n_2,n_3}$
    $(F^2)_{n_1 N_3 + n_3, k} := A^1_{n_1,k} A^3_{n_3,k}$
  - $r = 3 : (Y^3)_{n_3, n_1 N_2 + n_2} := Y_{n_1,n_2,n_3}$
    $(F^3)_{n_1 N_2 + n_2, k} := A^1_{n_1,k} A^2_{n_2,k}$

To understand how $F's$ and the $Y's$ variables have been computed in Algorithm 5, some examples have given below for each variable.

Let us see one example on how to calculate F_1 while A^2 and A^3 are given:

$$A^2 = \begin{bmatrix} 3 & 4 & 7 \\ 5 & 6 & 8 \end{bmatrix}$$

$$A^3 = \begin{bmatrix} 11 & 14 & 17 \\ 15 & 16 & 18 \end{bmatrix}$$

In the example: $N_{A^2} = N_{A^3} = 2, k = 3$ i.e N is number of instances
so F^1 is a $(N_{A^3} * N_{A^3})xk = 4x3$ matrix:

Algorithm 5 Alternating Least Squares algorithm for L2-MRMF

procedure MRMF-ALS
input: Z^1, Z^2, Z^3, k, *weight constants* $\alpha_{z^1}, \alpha_{z^2}, \alpha_{z^3}, \alpha_{z^{1\_bool}}, \alpha_{z^{2\_bool}}, \alpha_{z^{3\_bool}}$, *regularization constants* $\lambda_{a^1}, \lambda_{a^2}, \lambda_{a^3}, \lambda_b$
  $A^1 \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $A^2 \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $A^3 \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $B \sim \mathcal{N}(0, \sigma\mathbf{I})$
  repeat
    $A^1 \leftarrow \left(\alpha_{z^1} Z_1 B + Y^1 F_1\right) \left(\alpha_{z^1} B^T B + F_1^T F_1 + \lambda_{a^1}\mathbf{I}\right)^{-1}$
    $A^2 \leftarrow \left(\alpha_{z^2} Z_2 B + Y^2 F_2\right) \left(\alpha_{z^2} B^T B + F_2^T F_2 + \lambda_{a^2}\mathbf{I}\right)^{-1}$
    $A^3 \leftarrow \left(\alpha_{z^3} Z_3 B + Y^3 F_3\right) \left(\alpha_{z^3} B^T B + F_3^T F_3 + \lambda_{a^3}\mathbf{I}\right)^{-1}$
    $B \leftarrow \left(\alpha_{z^1} Z^1 A^{1.T} + \alpha_{z^2} Z^2 A^{2.T} + \alpha_{z^3} Z^3 A^{3.T}\right) \left(\alpha_{z^1} A^{1.T} A^1 + \alpha_{z^2} A^{2.T} A^2 + \alpha_{z^3} A^{3.T} A^3 + \lambda_B\right)^{-1}$
  until convergence
  return A^1, A^2, A^3, B
end procedure

$$F_1 = \begin{bmatrix} 3*11 & 4*14 & 7*17 \\ 3*15 & 4*16 & 7*18 \\ 5*11 & 6*14 & 8*17 \\ 5*15 & 6*16 & 8*18 \end{bmatrix}$$

Let us see another example on how to calculate Y^1, Y^2, Y^3. All three variables contain the same elements, yet they have different layouts.
For this example, Y is given. It contains the Y tensor items layout.

$$Y = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix}$$

$Y^1 : (Y^1)_{n_1, n_2 N_3 + n_3} := Y_{n_1, n_2, n_3}$

$$Y^1 = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix}$$

$Y^2 : (Y^2)_{n_2, n_1 N_3 + n_3} := Y_{n_1, n_2, n_3}$

$$Y^2 = \begin{bmatrix} 1 & 2 & 5 & 6 \\ 3 & 4 & 7 & 8 \end{bmatrix}$$

$Y^3 : (Y^3)_{n_3, n_1 N_2 + n_2} := Y_{n_1, n_2, n_3}$

$$Y^3 = \begin{bmatrix} 1 & 3 & 5 & 7 \\ 2 & 4 & 6 & 8 \end{bmatrix}$$

The second algorithm is SGD. SGD has been used in our previous semantic similarity CogALeX-V 2013 shared task as well. SGD is a stochastic approximation of the gradient descent optimization and the iterative method for minimizing an objective function as explained in Chapter 5. This study has been followed the same idea and procedure of the algorithm from the previous study. However it has different structure because of the number of matrices and structure. The SGD algorithm is shown

in Algorithm 6. This algorithm has designed and implemented by following the given visualization in Figure 6.1. In this algorithm, $A_i \in \mathbb{R}^k$ denotes i-th row of $A's$ (which also can be interpreted as the latent features of the word i). This definition is same for A_i^1, A_i^2, A_i^3. Analogously, B_j represents the j-th row of B.

For both algorithms, the matrices A^1,A^2,A^3 and B have initialized with random values. Now, the problem is to minimize the objective function. Both algorithms ASL and SGD have been used to optimize the objective function with respect to L2 loss function. This function is shown in Equation 6.5. The L2 loss function is basically minimizing the sum of the square of the differences between the target value and the estimated values, as explained in Chapter 4.

Both algorithms have been implemented and used in this study. However, when comparing ALS with SGD, the SGD algorithm implementation is simple and does not need large memory space as well. On the other hand, the ALS algorithm is slow during computation, and it needs large memory space when compared to SGD. Therefore, the experiments have proceeded with SGD algorithm for the reason of the computation time.

Algorithm 6 SGD for MRMF

procedure MRMF-SGD
input: Z^1, Z^2, Z^3, Y, k, *weight constants* $\alpha_{z^1}, \alpha_{z^2}, \alpha_{z^3}, \alpha_{z^{1\_bool}}, \alpha_{z^{2\_bool}}, \alpha_{z^{3\_bool}}$, *regularization constants* $\lambda_{a^1}, \lambda_{a^2}, \lambda_{a^3}, \lambda_b$
  $A^1 \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $A^2 \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $A^3 \sim \mathcal{N}(0, \sigma\mathbf{I})$
  $B \sim \mathcal{N}(0, \sigma\mathbf{I})$
  repeat
    $i \sim \mathcal{U}(1, m_1)$
    $j \sim \mathcal{U}(1, n)$
    ${A^1}_i \leftarrow {A^1}_i + \mu \left(\alpha_{z^1} \left(Z^1 - {A^1}_i B_j\right) B + \lambda_{z^1} {A^1}_i\right)$
    $B_j \leftarrow B_j + \mu \left(\alpha_{z^1} \left(Z^1 - {A^1}_i B_j\right) A^1 + \lambda_{z^1} B_j\right)$
    $i \sim \mathcal{U}(1, m_2)$
    $j \sim \mathcal{U}(1, n)$
    ${A^2}_i \leftarrow {A^2}_i + \mu \left(\alpha_{z^2} \left(Z^2 - {A^2}_i B_j\right) B + \lambda_{z^2} {A^2}_i\right)$
    $B_j \leftarrow B_j + \mu \left(\alpha_{z^2} \left(Z^2 - {A^2}_i B_j\right) A^2 + \lambda_{z^2} B_j\right)$
    $i \sim \mathcal{U}(1, m_3)$
    $j \sim \mathcal{U}(1, n)$
    ${A^3}_i \leftarrow {A^3}_i + \mu \left(\alpha_{z^3} \left(Z^3 - {A^3}_i B_j\right) B + \lambda_{z^3} {A^3}_i\right)$
    $B_j \leftarrow B_j + \mu \left(\alpha_{z^3} \left(Z^3 - {A^3}_i B_j\right) A^3 + \lambda_{z^3} B_j\right)$
    $i \sim \mathcal{U}(1, m_1)$
    $j \sim \mathcal{U}(1, m_2)$
    $k \sim \mathcal{U}(1, m_3)$
    ${A^1}_i \leftarrow {A^1}_i + \mu \left(\left(Y_{ijk} - {A^1}_i {A^2}_j {A^3}_k\right) {A^2}_j {A^3}_k + \lambda_{z^1} {A^1}_i\right)$
    ${A^2}_j \leftarrow {A^2}_j + \mu \left(\left(Y_{ijk} - {A^1}_i {A^2}_j {A^3}_k\right) {A^1}_i {A^3}_k + \lambda_{z^2} {A^2}_j\right)$
    ${A^3}_k \leftarrow {A^3}_k + \mu \left(\left(Y_{ijk} - {A^1}_i {A^2}_j {A^3}_k\right) {A^1}_i {A^2}_j + \lambda_{z^3} {A^3}_k\right)$
  until convergence
  return A^1, A^2, A^3, B
end procedure

$$\underset{A^1,A^2,A^3,B}{\arg\min} \quad ||Y - (A^1 * A^2 * A^3)|| \tag{6.5}$$

$$+ \alpha_{a^1}\frac{1}{2}||Z^1 - A^1B^T||_F^2 + \alpha_{a^2}\frac{1}{2}||Z^2 - A^2B^T||_F^2 + \alpha_{a^3}\frac{1}{2}||Z^3 - A^3B^T||_F^2 \tag{6.6}$$

$$+ \frac{\lambda_{a^1}}{2}||A^1||_F^2 + \frac{\lambda_{a^2}}{2}||A^2||_F^2 + \frac{\lambda_{a^3}}{2}||A^3||_F^2 + \frac{\lambda_b}{2}||B||_F^2$$

In the algorithms, the Y tensor contains the semantic relation information which is only the training set of the phrases. This training set of Y information has solely been used in order to optimize the matrices. However, only the distributional information has been used as input to test new phrases. Then the optimized matrices predict the semantic relation of the new phrases.

Equation 6.8 is used to predict new phrases semantic relation. This equation simply multiply the optimized context vectors of the phrase words. Then it predicts if they are semantically related or not. However, before the prediction, a threshold is learned on the training set to find the right split. Then Equation 6.8 output has been used to predict the phrases relation. If the phrases output result from the Equation 6.8 is above the threshold, they are semantically related, otherwise they are not related. The threshold is computed on the training set. For evaluation, the threshold has trained on the ten-fold cross-validation. In ten-cross-validation, 10% of the data is a test set and 90% of the data is a training set. Then the threshold value has been found to classify the phrases by taking the output value from Equation 6.8. The threshold training has followed the procedure which has been used in MRMF-SGD script which is written for the semantic similarity task 5. The MRMF-SGD for the semantic similarity was shown in Appendix A.2. From the Appendix, functions *rank* and *f1_of_rank* have been used to train a threshold. This procedure has been extended for the phrasal semantics task.

For the prediction, the three words of the new (test) phrases are expected to be in the optimized three matrices $A^1_{i,l}, A^2_{j,l}, A^3_{k,l}$ to simply use them in Equation 6.8 as an input and predict their relatedness. Otherwise, Equation 6.7 optimizes the new phrasal words representation by taking their context vector as an input. Then, A^{1_Test},A^{2_Test} and A^{3_Test} can be calculated through the fold-in. Finally, Equation 6.8 takes A^{1_Test},A^{2_Test} and A^{3_Test} as an input and predict their relatedness.

$$\begin{aligned} A^{1\_test} &= Z^{1\_test}B(B^TB)^{-1} \\ A^{2\_test} &= Z^{2\_test}B(B^TB)^{-1} \\ A^{3\_test} &= Z^{3\_test}B(B^TB)^{-1} \end{aligned} \tag{6.7}$$

$$\hat{Y}_{i,j,k} = \sum_l A^1_{i,l} * A^2_{j,l} * A^3_{k,l} \tag{6.8}$$

6.5 Parameter selection

This section explains the parameter selection of the SGD algorithm. In the study, a combination of learning rates, latent features size and regularization parameters with a wide range of values have been learned in order to find the best hyper-parameters value setting and optimize the matrices. A range in between $\frac{1}{\#\_A^1\_instances}$,

$\frac{1}{\#\_A^2\_instances}$, $\frac{1}{\#\_A^3\_instances}$ and $1 \cdot 10^{-7}$ have been considered to find the best learning rate of α_{a1}, α_{a2} and α_{a3} parameters value, respectively. The learning rate of α_y has set to 1 because Y is the tensor that we are building the model to the Y semantic similarity relation information and to classify the phrases relatedness. The μ parameter value has been learned with the rest parameters value by the list of values [0.1, 0.01, 0.001, 0.0001]. For the regularization constant λ ranges between $1 \cdot 10^{-2}$ and $1 \cdot 10^{-10}$ has been learned. Finally, a range in between 100 and 400 have been leaned for the latent features k.

The combination of the above parameters value has been learned on grid search to find the right values which optimize the matrices. From learning the parameter values, latent features size 100 has been showing a better performance with the μ learning rate 0,001. μ has been learned with a decay values [0.9, 0.95, 0.98, 1.0]. The highest result has been reported with a decay value of 0.95. In addition, the alpha parameters value has also played an important role. From the learned alpha parameters (α_{a1}, α_{a2} and α_{a3}), the parameters with the value 0.0000001, 0.00001 and 0.0001, respectively, have been giving the highest result with the μ 0.001 and decay 0.95.

6.6 Discussion and Conclusion

The model accuracy has been measured by F-score measure. After learning the different parameters value in between large value ranges, the highest reported result is 0,529 F-score. This result is reported from the SGD algorithm study. The ALS algorithm experiment could not proceed due to the very long computation time.

Category	Method	F-Score
Report Result (Baseline)	SVM	0.79
New Report	MRMF	0.529

TABLE 6.4: Results of the HsH and MRMF methods

Evaluating the semantic relation of phrasal words is the other semantic similarity task that MRMF method performance has been evaluated. However, this study is on phrasal semantics but the MRMF method did not perform as expected like the semantic classification task. As the result table, Table 6.4 shows, the MRMF model which has built on the SGD algorithm did not outperform the *HsH* method.

The main idea of the MRMF method is to integrate different sources of information effectively and optimize matrices for better representation. Although one of the reasons for the poor performance of the MRMF method can be the additional semantic similarity information of the words which have been integrated with the distributional information, it was not enough to improve the matrices with rich information. In addition, the phrasal semantics may need a different approach in order to integrate the matrices. Different parameter values and parameters have been investigated on the SGD algorithm. However, that did not increase the performance of the model above 0.529. The approach that has followed to build the model, has been showing positive performance on lexical semantics. But this approach did not give a satisfactory result for phrasal semantics. However, this approach can be seen from a different point of view, and more investigation can be done.

As a conclusion, the *HsH* method is still on the top rank to evaluate phrasal semantic task. MRMF method could not outperform the *HsH* method. However, the MRMF study on phrasal semantics needs more study. MRMF has shown good

performance on semantic classification and semantic similarity. Therefore, if it undergoes further studies on different algorithms and on different objective functions and/or different approaches, it may have a chance to improve the performance. Although, the further study has to be done in a very powerful system which can handle long computation and loop in a short time and also which has large memory space. One of the drawbacks of this study is the computation time because the matrices are dense and large. For accurate and efficient results, a high-speed system is required, such as systems with GPU.

Chapter 7

Applications

7.1 Overview of the Chapter

In this chapter, the three applications which have been introduced in our study are explained separately in more detail. Each application has a complete explanation under the following each section. Each section includes an introduction, related work, methodologies, result, conclusion and more specific details of each application.

The first application has been created to visualize the overall and main interest of German companies by using their website information. The application takes a German company website URI as an input and presents the information via concept cloud visualization. The concepts are taken from the Standard-Thesaurus Wirtschaft (STW)[1] Thesaurus of Economics. It contains vocabulary for all economic topics (ZBW - Leibniz Information Centre for Economics, 2014). STW has explained more in Chapter 3under the introduction and experiment sections. In the concept cloud, the colors of the concepts which have taken from STW, show the categories of the concepts in the thesaurus while the cloud layout is organized by the semantic proximity of the concepts. The semantic representations of concepts that are generated from DeWaC corpus have been used to compute the similarity between concepts. The distributional similarity that has used in the application is fundamentally different from the co-occurrence statistics which is often used to generate word clouds.

The second application has been created to automatically recognize and disambiguate the library of congress subject headings. We investigate the possibilities to extract the Library of Congress Subject Headings[2] from texts. The large number of ambiguous terms turns out to be a problem. However, the disambiguation of subject headings seems to have the potential to improve the extraction results.

The third application has created to automatically identify synonym relations in the Dutch Parliament's thesaurus. For indexing archived documents, the Dutch Parliament uses a specialized thesaurus. For good results of full-text retrieval and automatic classification, it turns out that it is important to add more synonyms on the existing thesaurus terms. In this work, we investigate the possibilities of finding synonyms for terms of the parliament's thesaurus automatically. To do this, we propose to use distributional similarity. In the experiments, we train and test a classifier using distributional similarity and string similarity with pairs of synonyms and non-synonyms. We have been able to classify 75% of the pairs from a set of 6000 word pairs correctly by using ten-fold cross-validation.

The three applications have been explained in the following sections in detail with respect to the order of the above three paragraphs brief explanation.

[1] http://zbw.eu/stw/version/9.02/about.de.html
[2] http://id.loc.gov/authorities/subjects.html

7.2 Constructing Concept Clouds from Company Websites

7.2.1 Introduction

People can have different reasons to search for information about a company. They might be looking for a product, a cooperation partner, a job, or an internship. In addition, for business political strategies, it is important to know what companies are active in a region and what they are doing (Garcia-Alsina, Wartena, and Lieberam-Schmidt, 2015). However, there are hardly any good sources with thorough and uniform information about companies. Most sources for information about companies are incomplete, outdated, or expensive and not easy to access. For example[3]: 1. Annual reports which describe their activities over the previous year, including details of their operations and headline financial results. Some companies also include their corporate social responsibility. 2. Internal documents which are not intended for public viewing (memos, emails, presentations, strategy documents or evaluations). On the other hand, almost every company has a website with information about products, activities, organization structure, jobs, etc. However, we need to visit each page of the website to know more in detail about the company; or one page of the website can summarize about the company, but it is very brief usually. This section explains a tool which extracts relevant information about a company from the company's website and presents the information in a concept cloud about the company.

The tool generates concept clouds from German company websites. The main idea of the visualization is designed to show the overall work and main interests of companies in detailed cloud-based information solely on their own web page. The concepts are taken from the STW Thesaurus of Economics. An economic thesaurus has been used to find important concepts which have expressed on the web pages of companies. All concepts related to products, economic sectors and several other categories are collected and presented in one concept cloud. In this cloud, semantically similar concepts are grouped together. The presented concept cloud tool consists of three components: keyword extraction from company websites, computation of the semantic similarity between the concepts, and the generation of a word cloud.

Word clouds are used for the visual representation of texts. The font size and color of a word show the importance and the category of the word respectively. The position of a word in the cloud can be arbitrary or reflect the relation of the word with other words. In our concept cloud, the colors of the concepts show the categories of the concepts in the thesaurus while the cloud layout is organized by the semantic proximity of the concepts. Concepts that belongs to the same category (e.g. commodities, economics, sectors, etc) are marked by the same color.

Semantic representation has been used to compute the similarity between concepts. Each concept has been represented by the context vector which has been constructed using DeWaC corpus. The distributional similarity is fundamentally different from the co-occurrence statistics which are often used to generate word clouds. This has explained in detail in the following sections.

The rest of the section which discusses the *Constructing Concept Clouds from Company Websites* has been organized as follow. Section 7.2.2 discusses the related work of the keyword extraction and the word cloud. Section 7.2.3 explains the methods that they have proposed for the challenge. The tool is explained in Section 7.2.4. Finally, Section 7.2.5 concludes this section (*Constructing Concept Clouds from Company Websites*).

[3] https://corporatewatch.org/sources-of-information-companies/

7.2.2 Related work

Keyword Extraction

Automatic keyword extraction has been studied since the middle of the previous century (Salton and Buckley, 1988). In 1972, Spärck Jones (reprinted as Spärck Jones (2004)) proposed a weighting for the specificity of a term. Spärck Jones (2004) argued that terms should be weighted according to collection frequency. Then, it matches on less frequent. This has become known as *term frequency–inverse document frequency* (*tf.idf*). *tf.idf*[4] is an information retrieval technique which weighs a term's frequency (TF) and inverse document frequency (IDF). Each term has their respective TF and IDF score. The product of the TF and IDF scores of a term is called the TF*IDF weight of that term. However, it turns out not to be the only criterion for keywords. More features can be found that indicate whether a term is suited as a keyword or not. These features can be used in a supervised machine learning setting to learn how to distinguish keywords from non-keywords together with the relevance weight of the term. This approach was proposed by Frank et al. (1999) and Turney (2000) for keyword extraction.

The other study is Wang, Liu, and Wang (2007) who provided keyword extraction algorithm based on WordNet and PageRank. Wang, Liu, and Wang (2007) use PageRank to determine the most central words in the graphs which are constructed with the WordNet relations between the potential keywords. WordNet is a large lexical database of English which nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), as discussed in our semantic classification study Chapter 4. In Wang, Liu, and Wang (2007)'s study, WordNet[5] has used as an undirected weighted graph, which defines synsets as vertices and relations of synsets as edges and assigns the weight of edges by the relatedness of connected synsets. Then, the PageRank has applied on the graph to prune the graph by doing some word sense disambiguation. PageRank is an algorithm which is used widely by search engines such as Google, for ranking the importance of website pages. PageRank views the web as a directed graph. But, in the Wang, Liu, and Wang (2007) study, the WordNet graph is undirected. Therefore, they have modified the PageRank equation, in which, case the out-degree of a vertex is equal to the in-degree of the vertex. The original PageRank formula is shown in Equation 7.1, and while the modified formula is shown in Equation 7.2. Finally, the PageRank has applied again, after pruning the original WordNet graph, to extract the keywords. Alternatively, the set of possible keywords can be restricted by the terms of a thesaurus. This approach is followed by Campos and Romero (2010), who use a thesaurus in combination with Bayesian statistics to suggest keywords. Bayesian Networks is a way of representing the structure of the probability distribution of variables. Campos and Romero (2010) followed this approach to build a model which classify documents by automatically generating an ordered set of appropriate descriptors extracted from a thesaurus. The Bayesian Networks has used in the method to model the thesaurus and uses probabilistic inference in order to select the set of descriptors with a high

[4] https://www.elephate.com/blog/what-is-tf-idf/
[5] https://wordnet.princeton.edu/

posterior probability of being relevant given the document to be classified.

$$PR(V_i) = \frac{(1-d)}{N} + d \cdot \sum_{p \in In(V_i)} \frac{PR(V_j)}{Out(V_j)}$$

Where:
N is the number of pages
d is called the damping factor and it is an arbritary weighting factor.
$PR(V_i)$ is the PageRank of vertex V_i.
$In(V_i)$ set of vertices that point to V_i,
$Out(V_i)$ set of edges going out of vertex V_i. (7.1)

$$PR(V_i) = (1-d) + d * \sum_{j \in In(V_i)} \frac{PR(V_j)}{|Out(V_j)|} \quad (7.2)$$

Gazendam, Wartena, and Brussee (2010a) have introduced a new weighting schema *tf.rr* which uses both a document and thesaurus information, and rank thesaurus terms to be used as keywords for a document. The schema uses frequency information of a term from a document, and the number of realized thesaurus relations between the thesaurus terms found in the specific document. The *tf.rr* uses only the thesaurus as a frame of reference. Equation 7.3 shows the formula *tf.rr* that Gazendam, Wartena, and Brussee (2010a) used in order to rank the thesaurus terms for a document by assigning weights to each term. In Gazendam, Wartena, and Brussee (2010a) terms which have a large number of relations, acquire higher weights in order to promote central concepts/keywords. Thus, the keywords central concepts have been extracted by following the related approach from the thesaurus for our tool as well.

$tf.rr(t,d) = tf(t,d)rr(t,d)$
where:
$tf(t,d) = 1 + log(n(t,d))$
$rr(t,d) = 1 + \mu r_1(t,d) + \mu^2 r_2(t,d); \mu = \frac{\alpha}{avlinks}; \alpha = 0.5$
$r_1(t,d)$; the number of realized relations at distance 1
$r_2(t,d)$; the number of realized relations at distance 2
$n(t,d)$; the number of occurrences of t in d
$\alpha$is a damping factor
avlinks is the average number of relations a term has in the thesaurus (7.3)

Word Clouds

Clouds of words have already been popular for several years for presenting user-generated tags. These clouds are called tag clouds. Word clouds usually denote similar graphical representations of words which are extracted from one or more texts. The words can be selected based on their frequency or more advanced keyword extraction algorithms which can be used. Usually, the font size of each word corresponds to the frequency or importance of that word. A number of popular

tools, such as Wordle Viegas, Viegas, Wattenberg, and Feinberg (2009), are available for constructing word clouds from text.

The words in a word or tag clouds can be ordered alphabetically or the positioning of the words can be optimized to fill all space (Kaser and Lemire, 2007). Hassan-Montero and Herrero-Solana (2006) proposed a layout for tag clouds based on the similarity of tags. Their algorithm is based on clustering tags and arranging tags on lines. Later, Gambette and Véronis (2010) proposed a tree representing an agglomerative hierarchical clustering. The disadvantage of this approach is that only the similarity of one word to the other word can be considered. This means, words with two aspects will not be placed between the words to represent these aspects, but they will just be tied towards one of the two words. In practice, highly similar words often end up on completely different branches of the tree.

Jacomy et al. (2011) proposed an algorithm for a graph layout based on the centrality of nodes and the strength of the edges. By a gradient descent method a local optimum is found for the position of all nodes in a two-dimensional space. A similar algorithm is proposed by Cui et al., 2010 as well. Cui et al., 2010 also use distributional similarity for word cloud layout on English words from a collection of documents.

In order to compute the similarity of words in a word cloud, word co-occurrence is most commonly used (Hassan-Montero and Herrero-Solana, 2006; Gambette and Véronis, 2010; Hirsch and Tian, 2013; Abulaish and Anwar, 2013). According to the distributional hypothesis (Harris, 1954), words that frequently co-occur do not have a similar meaning. E.g. the words *paper* and *review* co-occur frequently but have different meanings. The words *paper* and *article*, on the other hand, are almost synonyms but usually do not co-occur. Synonyms and other words with similar meaning are rather found by comparing the contexts in which they occur. The resulting similarity is called *distributional similarity*. In our study, we have proposed to use distributional similarity for the semantic proximity of the words in the concept cloud.

7.2.3 Method

The task which has been considered is visualizing German company websites using keywords and the keyword's context vector for the layout. Vectors of co-occurrence features have been used to represent each keyword and compute the cosine similarity between the keywords. The following subsections explain the consecutive steps that have been followed to visualize the website in detail.

Harvesting of the websites

For harvesting the websites, the same procedure has been used as in Wartena and Garcia-Alsina (2015). Wartena and Garcia-Alsina (2015) investigated the possibilities to extract information from websites of companies, and classify companies and other organizations using the website information. As a result, the websites are harvested using crawler4j[6]. We limited the number of pages to be retrieved to 120. Since it has crawled in a breadth-first search way, the most important information from the highest levels in the site hierarchy has been found. Breadth-first search is an algorithm for searching tree data structures. It starts at the tree root, and explores all of the neighbor nodes at the present depth prior to moving on to the nodes at the next

[6] http://code.google.com/p/crawler4j/

depth level. If there is a deep branch with a lot of specific information on one topic, it thus will be excluded.

Sometimes the crawling is not successful. The main reasons for this are: 1) The site forbids crawling, and 2) the site of the company redirects to another domain; since the crawler only follows links to the same domain, nothing will be retrieved in these cases.

Any boilerplate removal has not been used since it turned out that in many cases, essential information is removed by most algorithms. For example, a list of products or departments is often given as a menu. Thus, in many cases, important information will be removed. Moreover, companies do not use advertisements from other companies on their sites, so all the information on the site is usually relevant.

If all pages have been gathered from one site in order to count the words that have been used on that site, the counts can be distorted by headers, footers, and menus which appear on almost every page. In order to solve this problem, each website has been rendered as plain text. Next, the text has been divided into smaller fragments. Initially, the text was split on each blank line. Next, all consecutive fragments consisting of less than 50 characters were merged. Finally, all duplicate fragments were removed.

The tool has been developed for German company websites. For websites which are hosted in Germany, an imprint is obligatory. It is called an *Impressum* in German. An impressum is the term given to a legally-mandated statement of the ownership and authorship of a website. An impressum is included in books, newspapers, and magazines as well, which have published in Germany and certain other German-speaking countries, such as Austria and Switzerland. These imprints usually contain a lot of text that is not related in any way to the activities of the company. We use a number of heuristics to find a page or a section that is the imprint. We mark all fragments from the imprint and do not use these fragments for counting keywords.

Concept Extraction

The tool starts the process by extracting keywords from the company's website. The goal of keyword extraction is to find a few numbers of characteristic terms, usually just 3 or 5 or 10 words at most. In a word cloud, much more words can be presented. Thus, the exact ranking is less important. Moreover, the importance of a word in a network structure is visible in the word cloud. It does not have to be computed. A term that has many closely related terms in the cloud forms a cluster of words around a central concept. Therefore, using a central concept becomes important in the visualization, since the website text consists of many words. Thus, the terms have been annotated by a concept. After annotating each related term via a central concept, the frequency of the concepts has been considered in order to take concepts for the cloud.

The STW Thesaurus for Economics[7] has been used as a source to extract concepts for the words that are searched in the web texts. For the annotation of the terms with thesaurus concepts, we use a Gate pipeline (Cunningham et al., 2002) to perform language identification, tokenization, sentence splitting, and lemmatization. Concepts in the thesaurus were searched for with Apolda (Wartena et al., 2007). If a word is ambiguous and represents two different concepts, the word is annotated with both concepts. In order to increase the number of concepts, the enriched version of the STW which has described in Wartena and Garcia-Alsina (2015) has been used.

[7] http://zbw.eu/stw/versions/latest/about.en.html

Finally, the concepts are counted. The 40 most frequent concepts are passed to the visualization component along with their frequency and their main STW class.

Similarity Computation

As Chapter 2 explained, in distributional semantics, the meaning of a word is represented by a vector of context features. As context features of a word, co-occurrence data with the neighboring words in a large text corpus are used. In the end, we do not measure how often two words co-occur, but we check whether they occur in similar contexts or not.

To construct the context vectors of the concepts, we have followed the procedure which has been explained in Chapter 2 Section 2.5.1. However, since the concepts are German, we have used DeWaK corpus instead of UkWaK.

First, it has to be determined what words are used as context features, i.e. what words co-occurrence statistics have to be computed. Generally, it is found that mid-frequency words are the most effective. We have found that including all words in the frequency range from 4000 to $1 \cdot 10^6$ in the DeWaC Corpus is a good compromise between optimal results and acceptable storage and computing efforts. Then, each word is now represented by a vector of 16 565 features.

The labels used for the concepts in the STW Thesaurus for Economics are sometimes very frequent words but often infrequent words. It was shown in various studies (Weeds, Weir, and McCarthy, 2004; Wartena, 2014) that most similarity measures have a strong tendency to assign small distances to frequent words. In order to overcome these problems, the frequency adjusted cosine similarity has been used from Wartena (2014) study. The adjusted cosine similarity gives us the difference of the actual cosine and the cosine. We would expect on the base of the frequency of the words in the corpus, if the cosine is smaller than the expected adjusted cosine, this will be negative. We construct a cloud in which all nodes distance are measured by edge weight if their adjusted cosine is positive. Now, the edge weight is the cosine between the nodes.

7.2.4 Visualization

FIGURE 7.1: Concept cloud for a wood pulp ("Zellstoff") manufacturer

To visualize the concept, the frequency has been used in order to determine the font size and the category from the STW Thesaurus has been used to determine the color. For the cloud layout, the *Force Atlas* algorithm has been used from Gephi which is an open source software for graph and network analysis (Bastian, Heymann, and Jacomy, 2009; Jacomy et al., 2011).

Figure 7.1 shows an example of the concept cloud for a wood pulp manufacturer company website. In the figure, the color shows the category of the concepts. The purple concepts are commodities, blue is used for concepts from business economics, red for general economics, light green for economic sectors and green for all other categories. From the figure, we can see that related concepts are in most cases adjacent. However, we do not see a clustering into different areas. We can consider this as an advantage since a clear clustering often is not possible and concepts are related to more than one area. From the cloud, we can see at one glance that the main product of the company is wood pulp (*Zellstoff*). Furthermore, we can see that the company pays attention to quality management, learning on the job and environmental protection as well.

In addition, Figure 7.2 shows the concept cloud of an "Orthopaedic" (*Medizinische behandlung*) shoe manufacturer company website as an example. The cloud shows that the main product of the company is orthopaedic shoes. In the cloud, the related words are in most cases adjacent. If we see the related word's position in the cloud, words that are related to medicine are on the bottom; words that are related to therapy, sports, and rehabilitation, are in the center; and words that related to business and production are on the top. The concept cloud shows that the company promotes some sports activities and psychotherapy too.

FIGURE 7.2: Concept cloud for an Orthopaedic ("Medizinische behandlung") shoe manufacturer

7.2.5 Conclusion

We have built a tool which is consisting of three components. Starting from a URL of a Germany company website, German texts are collected and analyzed from their website. Then, central concepts are extracted from these texts and visualized in a concept cloud. The semantic proximity of the concepts in the concept cloud is based on the semantics of the concept labels computed by statistical means from a large text corpus. The tool can easily be executed in one batch process.

For future research, we have recommended the improvement of the similarity computation. An interesting possibility that one can investigate is to annotate the

whole text corpus with extracted concepts from the STW. Then, it is possible to compute the distributional semantics of the concepts instead of the similarity of the preferred labels of these concepts. Since some concepts can have a large number of different labels, it has been expected to get quite different results.

Besides concepts, it is possible to extract address information and links to other companies from the web texts. The concept clouds eventually can be used to present the search results.

7.3 Automatic Recognition and Disambiguation of Library of Congress Subject Headings

7.3.1 Introduction

Library of Congress (LoC) Subject Headings (LCSH) constitute a huge collection of terms that can be used for indexing. LCSH should not be confused with the much smaller and more structured Library of Congress Classification system. Yi and Chan (2010) gave a structural analysis of LCSH. We investigate possibilities of assigning LCSH terms automatically by extracting them from the abstracts of records which have been manually annotated with LCSH. A specific problem in LCSH is the ambiguity of many labels. LoC subjects may have a number of variants or alternative labels. Two subjects can have the same word as a variant. If this variant occurs in a text, it is not clear which subject should be used for indexing. E.g. the term *plants* is a label of `lcsh:sh85102839`[8] (green-growing things) as well as of `lcsh:sh85046823` (manufacturing facility). Usually, different LCSHs for a term refer to different aspects of the same concept. E.g. `lcsh:sh85000800` (adaptation) denotes the adaptation of plants and animals to ecosystems, while `lcsh:sh85000892` refers to behavioral adaptation in psychology.

Given the large amount of LCSH concepts, we cannot use classification, but we should rather use keyword extraction. Pouliquen, Steinberger, and Ignat (2003) distinguish between conceptual thesauri and natural language thesauri. In the first case, they argue that keyword *extraction* is not suitable since most concepts never will be found in the literal form in texts. Here we should use keyword *assignment*. LCSH clearly has characteristics of a conceptual thesaurus. For keyword assignment Pouliquen et al. build signatures for each concept, where a signature is basically a vector of word weights. As for classification, the need for the amount of training material for each concept is the bottleneck of the approach. Wartena, Brussee, and Slakhorst (2010) use distributional similarity between potential keywords and abstracts to rank candidates. Below we use a similar approach to disambiguate ambiguous keywords.

The remainder of this section is organized as follows. In Section 7.3.2 we present the data which have used in the study, and how it has constructed. The related work is discussed in Section 7.4.3. In Section 7.4.4 we introduce the data used for the experiment. Section 7.4.5 describes the experiment and Section 7.4.6 gives the results. Finally, we reflect on the practical implications of the experiment and discuss further steps in Section 7.4.7.

7.3.2 Data

The LCSH collection ([9]http://id.loc.gov/download/) consists of 414 355 subject headings. 162 569 of these concepts are pre-combined concepts, e.g. `lcsh:sh94004310` with the label *Voyages and travel – Mythology*. The labels with dashes will never occur in running texts. Thus we removed these labels but kept eventual variants without dashes.

There are 3605 headings that are meant to be used as subdivisions of the main headings. We removed these headings, as well as all Children's Subject Headings

[8] We use the namespace `lcsh` for `http://id.loc.gov/authorities/subjects/`

[9] `\unskip\penalty\@M\vrulewidth\z@height\z@depth\dp`

	Records	Subject headings	Unique subj. head.	Pre-combined	Subdivisions
Original	12543	73528	11546	45857	104
Mapped	9982	48790	4259	30289	108
Combined	14267	89440	13428	47576	212

TABLE 7.1: Occurrences of subject headings in our data set.

and all inverted labels were removed. 497 427 labels now are left. Since many headings are in plural, we have lemmatized all headings and added singular forms which did not lead to ambiguity. As a result, we have added 75 270 labels.

Many (ambiguous and non-ambiguous) headings have some additional terms which specify the intended sense. These terms are put in brackets as a part of the label, like e.g. *Taxis (Biology)* and *Taxis (Vehicles)*. We have removed these disambiguating terms, of course increasing the number of ambiguous labels. Now there are 15 661 ambiguous labels. Most ambiguous labels (12 010 to be precise) just have two senses, and over 1 000 of them have three meanings. The most extreme cases are the words *cooking*, *suites* and *concertos* that each have over 400 senses. Finally, there are highly frequent words with a very infrequent meaning: e.g. the word *in* is a label for the subject `lcsh:sh85069880` (Confucian Philosophy). Thus, we have removed labels that have been found in a common stop word list.

For keyword assignment, we use data from the catalog of 200 scientific libraries of the German Länder Bavaria, Brandenburg and Berlin, called B3Kat. The catalog can be downloaded or queried over a SPARQL Endpoint[10]. We have retrieved about 16 000 records in English, with a title, a subtitle and an abstract of at least 200 characters. After filtering out records with French and German abstracts, 15 629 records have remained.

Next, we have retrieved for all records all available LoC subject headings and all keywords from the authority file of the German National Library, the *Gemeinsame Normdatei* (GND). Using the mapping from GND to LCSH which is a part of the GND, we have managed to add more subject headings. The number of occurrences of original and mapped subject headings with their number of pre-combined and subdivision headings is given in Table 7.1. The table contains the information of combined data as well, which is either an LCSH term or an LCSH term mapped from a GND term.

7.3.3 Experimental Setup

In order to disambiguate the ambiguous headings, we have compared the words in the title, subtitle and abstract of a record with the "glosses" of each potential subject. As a gloss for the subject, we have been taking all preferred and alternative labels of the subject and all disambiguating terms which have been used in the labels. The subject with the largest overlap between the subject of the gloss and the abstract is selected. In many cases, the text and the words related to the correct subject heading do not have overlap. If the words are not the same but at least semantically related, we can use distributional similarity to find the most likely subject heading. Co-occurrence vectors present the meaning of a word quite well (see e.g. Kiela and Clark (2014)). Thus, for each word, we have built a vector of co-occurrence frequencies with 17 400 mid-frequency words from the UkWaC Corpus by following

[10] `https://lod.b3kat.de/doc/sparql-endpoint/`

	#predict-ions	Original		Mapped		Merged		
		Recall	Precis.	Recall	Precis.	Recall	Precis.	F_1
Unamb. headings	12,6	0,160	0,0535	0,195	0,0585	0,219	0,0728	0,109
All headings	37,0	0,180	0,0225	0,210	0,0236	0,242	0,0302	0,0537
+Pre-combined	43,3	0,182	0,0210	0,213	0,0217	0,245	0,0279	0,0501
Disambiguated	18,1	0,166	0,0374	0,192	0,0388	0,224	0,0501	0,0819

TABLE 7.2: Average results, evaluated on original, mapped and merged annotations

the framework in Chapter 2 Section 2.5. We have computed the average of all co-occurrence vectors of all words from the gloss. For this average, we give the words which have been found in the scope notes with a smaller weight, instead of the labels themselves. We have also computed the average co-occurrence vector of all words in the text. Now we can compute the cosine between the average vectors and select the subject heading with the largest cosine.

To extract the labels, we lemmatize the text using the Natural Language Toolkit[11] and collect all occurrences of labels in the words and in the lemmata. We have been doing 4 runs. In the first run, we have only considered labels without any disambiguation information. In the second run, we have included all these labels, but without doing any kind of disambiguation. The third run is identical to the second one, but we have added all pre-combined subject headings for which each of the components have been found. The last run again is the same as the second one, but now we have used the disambiguation to select the most probable concept for each ambiguous term. For words which do not have enough co-occurrence values in the UkWaC Corpus, we choose the subject without disambiguating terms.

7.3.4 Results and Conclusion

Table 7.2 gives the average results for the extraction of subject headings for all 14 267 records. We have evaluated against the original LCSH annotations (12 543 records), against LoC headings obtained by mapping from GND-terms (9 982 records) and against the merged annotations (14 267 records).

It becomes clear that extracting LoC subjects from text is a very challenging task. We have over 400 000 subjects. The small test set of 14 267 records uses almost 90 000 different subjects. These subjects range from abstract classes like *science* to the name of specific buildings on the fairground in Hanover (*Halle 13*) or to complex concepts like *Behavior therapy for children*). It should be noted that keyword extraction often yields a low recall, even if much smaller vocabularies are used. Gazendam, Wartena, and Brussee (2010b) reach an average recall@5 of 0, 23 using a thesaurus of 3800 concepts, and Medelyan and Witten (2005) report a recall@5 of 0, 197 using the Agrovoc-thesaurus (16 000 terms). Kim et al. (2013) report on a shared task for keyword extraction from scientific articles in which 27 systems were evaluated. In this task, keywords were freely assigned by authors and readers. The highest recall@5 was about 0, 13; the highest recall@15 was 0, 27. Paynter (2005) does not use keyword extraction but assigns LCSH which is found in similar documents. As a result, he reaches a precision of 0, 19 and recall of 0, 21. The overlap between the original LCSH, and the mapped LCSH is also very low. The average Jaccard coefficient is 0.18 for 8258 records with both original and mapped LCSH annotations. When

[11] http://nltk.org/

we treat the original annotations as our gold standard and the mapped headings as predictions, we have a recall of $0,15$ and a precision of $0,27$.

We see that the best F-score is reached when we do not consider ambiguous labels at all. If we do not exclude ambiguous labels from the prediction, we get best results when we disambiguate the labels. However, we get the optimal result when we completely exclude all ambiguous lables from the prediction.

7.4 Automatic Identification of Synonym Relations in the Dutch Parliament Thesaurus

7.4.1 Introduction

The Information Service of the Second Chamber of the Dutch parliament *(Tweede Kamer der Staten-Generaal)* archives and indexes of documents produced in the parliamentary process, and other documents that are possibly relevant to the parliament. The parliament is used for indexing a special thesaurus that covers all topics relevant to the society. The Dutch parliament is investigating additional alternatives to make information available via full-text search, automatic indexing and automatic classification. For full-text retrieval and automatic classification, it turns out that adding more synonyms to the existing thesaurus terms for a good result is important.

In this work, we investigate the possibilities to find synonyms for terms of the parliament's thesaurus automatically. To do this, we propose using distributional similarity. Distributional similarity assigns small distances to terms that are semantically related. However, this relatedness does not correspond to any traditional semantic relation such as synonymy. Terms that are related by distributional similarity might be synonyms, or antonyms or other relation of the words as well. In order to see whether distributional similarity can be used to distinguish between related and unrelated terms in this thesaurus, we conducted an experiment on 6000 pairs of terms from the parliament thesaurus. Half of the pairs consist of related words and the rest half of the pairs are not related. Regarding these pairs, we train and test a classifier that distinguishes pairs of related from pairs of unrelated words.

The remainder of this section is organized as follows. In Section 7.4.2 we present the Parliament's Thesaurus and its role in the information processes of the Dutch parliament. The related work is discussed in Section 7.4.3. In Section 7.4.4 we introduce the data used for the experiment. Section 7.4.5 describes the experiment and Section 7.4.6 gives the results. Finally, we reflect on the practical implications of the experiment and discuss further steps in Section 7.4.7.

7.4.2 Information Processes in the Dutch Parliament

Every day, the Dutch parliament processes and produces many documents. In order to make the information available to the users – i.e. the internal users of the Parliament and also the Dutch citizens – documents are enriched with a large amount of metadata, among which subject terms from a controlled vocabulary. A parliamentary thesaurus (*Parlementsthesaurus*) has been used for many years as a source for subject indexing of those documents that are produced in the parliamentary process, as well as other documents that are relevant for the parliament such as reports, articles in journals and newspapers, and interviews.

Indexing

In the past, subject metadata was the only source of information about the subject of documents. Today, all parliamentary documents' full text is available and also searchable online. However, in order to make documents accessible in an organized and coherent manner, subject metadata are still a valuable source of information. Using subject terms from a controlled vocabulary such as a thesaurus may add value to the result of information seeking. In addition, both the fraction of relevant documents that are retrieved (recall) and the fraction of retrieved documents that are

relevant (precision) can also enhance under certain circumstances. Furthermore, subject keywords have been found to be valuable tools for enhancing the results of keyword searching in Online Public Access Catalogs (OPACs) (Gross and Taylor, 2005). Gross and Taylor (2005) found that more than one-third of records retrieved by keyword searches would be lost without controlled subject keywords. This finding was recently replicated after the addition of automated enriched metadata such as summaries and tables of content (Gross, Taylor, and Joudry, 2015).

Searching documents on a specific subject is possible via entering one or more subject terms from the thesaurus directly. However, finding only documents that have been indexed with a combination of the subject terms is in question. Subject terms are used in the presentation of information, e.g. as a basis for word clouds and indicate areas of interest of parliamentarians using the cloud. In order to enhance the results of the full text search process, the thesaurus is linked to the search engine e.g. by the use of synonyms. In principle, this applies to the semantic relations as well. However, including semantic relations in the full-text search may affect the relevance of the search results.

Subject indexing is performed manually by the information officers of the parliament (First and Second Chamber). This, of course, is a rather laborious process. In order to improve the efficiency, the consistency and the quality of indexing, efforts have been taken to automate subject indexing. Various forms of language technology are used for this purpose. However, until now this has not lead to a useful application of the technology. It is believed that this is due to the breadth of the thesaurus, the large number of subject terms and the specific features of parliamentary documents. In order to improve the usefulness of the thesaurus for new applications, one direction is to automatically add more synonyms to the thesaurus, especially frequently used words in the parliamentary context.

The Parliament Thesaurus

The origin of the current thesaurus occurred in the 1980s when the first steps were taken to develop a controlled vocabulary for the Dutch parliament. Nowadays, the *Parlementsthesaurus* is a large polyhierarchical thesaurus that has been built around on a number of main subjects or policy areas. A wide range of policy areas is covered from health care and education to environmental planning and agriculture. The thesaurus consists of $>$ 4000 descriptors and $>$ 6000 non-descriptors, along with their semantic relations, synonyms and definitions (scope notes). Semantic relations include hierarchical relations (broader, more general, and narrower, more specific concepts), and also concepts that are otherwise related ("associative" relations).

Maintenance of the thesaurus is carried out by a thesaurus manager in consultation with a number of thesaurus editors and information officers who are specialized in one or more policy areas and who are using the thesaurus in indexing. The nature of this process enhances the substantive quality of the thesaurus. However, it also makes the thesaurus less dynamic. In the present time, it is more desirable that it takes less time to include new subject areas and new terminologies. Therefore, the information office of the parliament is looking for a way to improve the dynamic properties of the thesaurus while maintaining the substantive quality.

7.4.3 Related Work

The construction of thesauri is often mentioned as a possible application of distributional similarity (Crouch, 1990; Crouch and Yang, 1992; Curran and Moens, 2002).

Nevertheless, there are only very few studies that concretely investigate the problem of inserting new terms into an existing thesaurus.

Witschel (2005) uses distributional similarity to find the right insertion position for new terms in a hierarchically organized taxonomy. Starting at the root, the taxonomy travels downwards as long as one child of the current node is more similar to the new concept than all its sibling nodes. However, even in a very small taxonomy, this method did not give very good results.

Meusel et al. (2010) use a web search engine and Hearst-patterns to find hyponyms and synonyms for new words in an existing thesaurus. In order to reduce the number of queries that have to be issued, the method is only applied to the 100 thesaurus terms which have the highest distributional similarity with the new term. They tested their method on two different thesauri. For the two-way classification between synonyms and non-synonyms, they got an accuracy of 98% and 85%. For the more difficult two-way classification between synonyms and hyponyms, they got an accuracy of 71% and 68%. Overviews of methods which are used in the more general problem of automatic thesaurus construction, are given by Biemann (2005) and Drumond and Girardi (2008), for example.

7.4.4 Data

In this section, we present the test sets that we have created and used for the experiment, as well as the corpus which has been used to construct the context vectors of all terms.

Word Pairs

There are two relations of the pairs which are not clear. These are: 1. how the relation between descriptors and non-descriptors in a thesaurus has to be described in traditional semantic terms, and 2. what type of semantic relation corresponds to distributional similarity. To clear up these relations, a new test collection has been developed. The goal of the test collection is to see whether distributional similarity can be used in order to describe the relation between non-descriptors and descriptors which are also known as the *use/use-for* relation in a thesaurus.

The test set consists of 6000 pairs of words. The 3000 pairs of words *(a,b)* where *a* and *b* are a label for the same concept. These pairs have been sampled randomly. For the negative pairs, a balanced quantity of easy and hard pairs has been included. Hard pairs are the pairs that are hard to distinguish from positive pairs, unlike easy pairs. To do so, 500 pairs of words have been selected that are labels for directly related concepts. Any type of relations specified in the thesaurus have been used. The next 500 pair of labels have been selected for concepts that are related by one intermedi- ate concept. The next 500 pairs have been selected for the concepts that have two intermediate concepts on the shortest path between each other in the thesaurus. The remaining pairs have been found in the same manner for a longer thesaurus distance each time.

Some examples of positive and negative pairs are given in Table 7.3.

Corpus

The distributional similarity between two words is basically a similarity in the distribution of those words in a large corpus. A corpus has been compiled to represent

term 1	term 2	intermediate nodes	synonymous
volksgezondheid (public health)	gezondheid (health)	0	+
regelgeving (regulations)	wetsvoorstel (bill, draft law)	0	+
krijgsraad (court-martial)	militair strafprocesrecht (military criminal procedure)	0	+
schilderij (painting)	beeldende kunst (visual arts)	1	-
volksuniversiteit (adult education center)	rijksuniversiteit (state university)	2	-
zwijgplicht (confidentiality)	politierechter (magistrate of a police court)	3	-

TABLE 7.3: Example of pairs of labels for the same and for different concepts and the number of intermediate concepts in the parliament's thesaurus.

the words by distributional information. The corpus has texts that are quite characteristic for the texts that are annotated in the Dutch parliament. Thus the words' meaning and use in the corpus are expected to be similar to the thesaurus.

As a base for the corpus, texts from `bestanden.officielebekendmakingen.nl` have been collected. The text contains all official publications from the Dutch government between the year 2010 and 2012. This site partially overlaps with the archived material from the parliament. Due to server and connection timeouts, the corpus does not contain all documents from more years. The raw corpus consists of 88,8 million words. Since many documents start or end with exactly the same formulations, only unique sentences have been kept. In the end, the corpus contains 47 million words.

For the computation of the distributional contexts of each word, the words of the corpus have been lemmatized using the TreeTagger with the parameter files included in the distribution, and all stop words have been removed. Then, the corpus is left with 40 million lemmata.

7.4.5 Experiment

In this section, a simple experiment has been described which trains a classifier on the set of positive and negative examples. As input features, one type of distributional similarity and string similarity have been concatenated to train the classifier. The following sections explain the two types of features (distributional similarity and string similarity).

Distributional Similarity

Firstly, feature vectors have been constructed for each word in order to represent them by a context vector. To construct the context vector, it is possible to tune different parameters like the window size of the context words, the frequency of the context words in the corpus to be considered as a context word, and many more. Though, the methodology that yields in most cases the best results (Bullinaria and Levy, 2007; Bullinaria and Levy, 2012; Kiela and Clark, 2014), have been used. This methodology has been explained in detail in Chapter 2 Section 2.5.1. However, the mid-frequency range between 200 and 1.10^6 occurrences in corpus has been used.

This is because the size of the corpus is small compared to the English corpus UkWaK that has been used to represent each word in other experiments of our projects. Then, a total number of 11 080 context features have been constructed to represent each word.

Finally, the cosine method has been used between the words' context vectors to measure the similarity distance. From the distance measures, cosine distance measure is the top from what we saw in our previous studies.

String Similarity

String similarity is one of the methods which has been used to find similarities between the words. This type of method can be captured easily with n-gram overlap or with edit distance. In our case, the trigram overlap has been used, since that turned out to be the most effective similarity measure. In addition, as far as we know, there is no combination of different string similarity measures which are better than just the trigram overlap.

For a word or string w=$w_0w_1....w_n$, the set of trigrams has been defined as $tg(w) = \cup_{i=0}^{n-2}\{w_iw_{i+1}w_{i+2}\}$. For two words w^1w^2 the trigram overlap has been defined as the Jaccard coefficient of their sets of trigrams: overlap(w^1, w^2) = $\frac{|tg(w^1)\cap tg(w^2)|}{|tg(w^1)\cup tg(w^2)|}$

Experimental Setup

For each pair of words, there are two features which are the cosine distance of the pairs using their context vectors and the trigram overlap of the pair of words. To train a classifier, Support Vector Machine (SVM) has been used with radial basis function (RBF) kernel. SVM classifies the pair of words into related and unrelated pairs.

LIBSVM has been used to learn the model and classify the word pairs which have represented by the two features. The hyper-parameters of the model have been tuned, using grid search. To find the best C parameter value, the numbers in between 0 and 20 in step 0.05 have been investigated. Ten-fold cross-validation has been used with stratified sampling for evaluation purposes.

7.4.6 Results

Table 7.4 shows the average results from ten-fold cross-validation. Distributional similarity and the trigram overlap have been seen as useful features to classify the pair of words. Clearly, using both features gives better results than using one of the features individually. The reached accuracy 0.75 is clearly better than the majority classifier (baseline), that would assign each pair to one of both classes. However, the result has more room for improvement.

The experiments which have been carried out by Meusel et al. (2010) are quite similar to our experiment. Since they use different thesauri and different test sets, it is not really possible to compare the results. Nevertheless, the results that they gave for the binary classification of synonym versus non-synonym terms are much better than our results. However, they use random word pairs for the non-synonyms, whereas we have deliberately selected difficult pairs, including hyponyms, co-hyponyms, and more. In fact, Meusel et al. (2010) also tested the classification of such difficult pairs in their classification of synonyms versus hyponyms. For this task, their results are slightly lower than our results.

Features	Accuracy
cosine	0.69
trigram overlap	0.72
both features	0.75

TABLE 7.4: Accuracy results of classification with ten-fold cross validation of 6000 pairs of labels for the Dutch parliament's thesaurus. Half of the pairs consist of labels of the same concept, while half of the labels are from different concepts.

7.4.7 Conclusion and Future Work

We have shown that distributed similarity can be used to model the relation between concept labels in a traditional thesaurus. In addition we have used string similarity and trained a classifier to combine both types of features. The classifier has classified about 75% of the pairs correctly for a dataset of 6000 related and unrelated word pairs which have been constructed for this task. Despite the fact that the proposed features are useful for the considered task, the classification is still far from the highest accuracy.

For future work, three different directions can be pursued. The first is studying more distributional similarity. For example, using the cosine for the similarity of the context vectors might not be a good choice in the given situation. Furthermore, the Hearst patterns can be integrated as one feature.

The second interesting question which has to be answered is the influence of the corpus which is used to construct the context vectors. While the influence of the corpus size on distributional similarity has been studied quite well, the small size of the corpus is known on the influence of the text selection. For the current task, it is possible to compare a neutral corpus with a specialized corpus within the same domain as the test set. Finally, the more realistic scenarios are extracting candidate terms from the corpus, and assigning these terms to the most likely concept and evaluating them manually.

Chapter 8

Conclusion and Future Work

This dissertation presents three major contributions to distributional semantics. The first contribution is showing that the supervised learning approach performs better than theunsupervised learning approach when it comes to building models for tasks in distributional semantics. The second contribution introduces multi-relation matrix factorization (MRMF) in distributional semantics as one of the supervised methods which has the potential to improve some models in distributional semantic by integrating different sources of information. The last contribution involves the introduction of three distributional semantic-based applications.

8.1 Contribution of the Dissertation and Future Work

We have considered two tasks to reach the conclusion our first contribution that is building models using the supervised learning approach outperforms the unsupervised learning approach in distributional semantics. These tasks are semantic similarity and semantic classification. These tasks have been analyzed by training a model on supervised and unsupervised learning approaches. Then, the model's performance has been compared with each other, and also with state of the art the unsupervised learning models performance which have been built for the tasks. This study contribution is shown in Chapter 3 and Chapter 4.

In Chapter 3, models have been trained on supervised and unsupervised learning methods to classify semantically-related pairs of words. As a result, the models which have built by the supervised learning methods outperformed the unsupervised models. In addition, these models have outperformed the state of the art unsupervised learning models as well.

In Chapter 4, models which classify words to their semantic category, have been studied on supervised and unsupervised learning approaches. Like, Chapter 3, the models which have trained on supervised learning methods have outperformed the models which have trained on the unsupervised learning methods to classify semantically-related words including state of the art models as well.

These two Chapters (3 and 4) studies have contributed toward reaching the conclusion and recommendation *i.e* to use supervised learning methods and train a model for tasks in distributional semantics. The supervised learning approach has been using methods such as support vector machine (SVM) and logistic regression (LR) to train the models for both semantic similarity and semantic classifications tasks. For the semantic similarity task, the methods have been taking the context vector of each pair of words as an input which has been constructed by simple mathematical equation (i.e multiplication) to trained the outstanding models which classify semantically related pair of words. For the semantic classification task, the methods have been taking context vector of each word as an input to build the outstanding models which classify semantically-related words.

As a conclusion, our first hypothesis has been proved correct by the experiments on the well known semantic classification and semantic similarity tasks. As a reminder, the first hypothesis that has stated in the introduction chapter (Chapter 1) is that *models which have trained using supervised learning approach to solve tasks in distributional semantic outperform the unsupervised learning approach*. However, if it is necessary, as a future work, this experiment can be studied further on additional tasks.

As a second contribution, MRMF has been introduced in distributional semantics. MRMF is mainly known for recommender systems. However, in this dissertation, the MRMF method has been studied to build a model for tasks in distributional semantics. One of the task that MRMF has applied and showed good performance, is the semantic classification task. On this task, the method has been used to build a model and categorize semantically-related words to their category. Then, the model performance has been compared with the supervised learning models that have trained on SVM and LG to classify semantically related words. These models have already outperformed the unsupervised learning models as stated in our first contribution. MRMF has outperformed these models which have been trained on the supervised and the unsupervised learning approach. This method study and output are shown in Chapter 4.

Basically, training a model with additional sources of information helps to improve a model's performance. MRMF has a smart way of integrating different sources of information to train a model. This smart integration is the key of MRMF to perform better comparing other supervised and unsupervised learning methods. As Chapter 4 shows that lexical and distributional information have been integrated using SMV and MRMF to train a model which classify semantically related words. Then, the model which have trained on MRMF have outperformed the rest of the models. Finally, chapter 4 has concluded that MRMF can easily and naturally integrate different sources of information, and train a better model.

The MRMF study has extended to more tasks such as semantic similarity and phrasal semantic tasks since it showed good performance on semantic classification task. The CogaLex shared task challenge has been used to study the MRMF performance on the semantic similarity task. The idea of CogaLex shared task is to propose a model which classify semantically-related pair of words. Thus, first, we have proposed the same supervised approach method *HsH-Supervised* that we have proposed in our previous semantic similarity study (Chapter 3). Then, MRMF has been studied to train a model for the task. As Chapter 5 shows that the model which has been trained using MRMF has outperformed that of the *HsH-Supervised*. However, there is more room to improve this model on this task. For example, the design of the method can be seen in a different approach or direction. The design of the method is the way that the matrices have visualized or structured to integrate the information in the method. In addition, different algorithms can also be investigated.

For the phrasal semantic task, the SemEval-2013 shared task 5a challenge has been used to investigate the performance of MRMF. The shared task has already a supervised learning method which is proposed in a literature. However, MRMF performance has been studied on this task, as well, since it showed performance improvement on the models of semantic similarity and semantic classification tasks. However, the method could not outperform the literature method performance. As Chapter 6 shows and explains in detail, it did not perform better compared to the literature method. As a conclusion of the experiments, the method performance could not go beyond lexical word tasks. MRMF has been showing performance improvement in the lexical word tasks model, such as semantic similarity and semantic

classification. However, the method did not exhibit the same performance for the phrasal semantic task. However, there are more ideas and approaches to investigate MRMF and improve the result of the phrasal semantic task model. This is explained later in the following paragraph.

As a conclusion of MRMF, the second hypothesis which is *multi-relation matrix factorization method outperform the tasks' model in distributional semantics which have trained by the well-known methods such as, support vector machine (SVM)*, has been proved correct on lexical term tasks. This hypothesis has been studied on lexical and phrasal semantic tasks. However, the experiment outcome on phrasal semantics was not like the lexical terms semantic tasks. After several experiments of MRMF on phrasal semantic task, it has decided to postpone the study of this task as future work because the method needs more investigation on different directions on this task. For example, the algorithms that have been used for this task can be modified. In other words, the statistical modeling of the algorithms needs to be investigated again or different approaches need to be used. Otherwise, different algorithms can also be investigated. In addition, more sources of information have to be integrated for the quality of the model and performance improvement. However, all of these have to be done with high speed and powerful system. One of the main challenges of this study is explained at the end of this chapter.

Finally, three applications which have built based on distributional semantics, have introduced. The three applications are an application: 1. for German company websites which visualize the overall concept of a company by concept cloud; 2. for the Library Congress subject headings which automatically recognize and disambiguate library congress subject headings; and 3. for the Dutch Parliament Thesaurus which automatically identifies a synonym relation in the Dutch Parliament Thesaurus. The main idea is to use distributional semantics in these kinds of applications for better performance. As it has been expected, the distributional semantics approach has helped to build the applications and showed better performance compared to other approaches such as simple string similarity. However, the applications have still more room for improvement to reach high accuracy and performance.

In general, in the dissertation, the two hypotheses from the introduction Chapter have proved correct. The performance of models which have trained in the supervised learning approach tasks in distributional semantics has been investigated. Then, it has shown that the first hypothesis is correct. The multi-relational matrix factorization has also been investigated for tasks in distributional semantics. Then it has been shown that the second hypothesis is correct. The success of MRMF in distributional semantics has been shown on lexical terms tasks. On these tasks, the method has performed better than the methods that have been used for the tasks using the well-known methods in the field such as SVM. Then, this method performance investigation has extended to phrasal words. However, the second hypothesis did not find correct on the phrasal words. Although, it is expected that the performance of this method on phrasal words might be improved by the future work recommendations.

One of the biggest challenges that we have been facing n this dissertation, was the MRMF experiments on the phrasal semantic task. This study's experiments have been taking really long times for computation. Therefore this study needs a very powerful system for a very satisfactory work and good results. In the study, matrices contain minimum 104 400 000 cells which need to be computed in an algorithm, and minimum 60 times in a single experiment with only one round of parameters value tuning. Therefore, we put this experiment in the future work for more investigation,

but it has to be using powerful systems such as systems with graphics processing unit (GPU).

Appendix A

Appendix for MRMF

The following appendices show the scripts of MRMF with the coordinate descent algorithm (MRMF-CD) and MRMF with the stochastic gradient descent algorithm (MRMF- SGD).

A.1 MRMF-CD for Semantic Classification

This appendix shows the MRMF-CD script which has used to train a model for semantic classification task. The script works for different datasets, but the following script is an example on the SC53 dataset. The script has been used when we have three different information to integrate on MRMF. In addition, it uses a coordinate descent algorithm in order to optimize the matrices.

```
from numpy import linalg as LA
import numpy as np
import os

########## Reading files ############
def data_reading(data):
    data_read_dict = {}
    data2 = open(data)
    for i in data2:
        i = i.strip().split('\t')
        i2 = i[1].split(',')
        data_read_dict[i[0]] = map(float,i2)
    return data_read_dict

############ Initialize the matrices ############
def random_matrices(m,n,c,l,k):
    U = np.random.randn(m,k)
    V = np.random.randn(n,k)
    C = np.random.randn(c,k)
    B = np.random.randn(l,k)
    return U,V,C,B

###### Structuring the Y and have only the class index in a vector. Eg.
    sc53 has 53 classes. In MRMF Y is a matrix with 53features#####
def construct_y(Y):
    y = []
    for i in Y.keys():
        y.append(list(map(float,Y[i])).index(1))
    return y

####### L2 loss objuctive function #############
```

```
def
    L2_loss(X,Y,Z,U,V,C,B,alpha_x,alpha_y,alpha_z,lambda_u,lambda_v,lambda_c,lambda_b,y):
    arg_min = float(alpha_x*0.5)*((LA.norm(X-np.dot(U,V.T)))**2) +
        float(alpha_y*0.5)*((LA.norm(Y-np.dot(U,C.T)))**2) +
        float(alpha_z*0.5)*((LA.norm(Z-np.dot(U,B.T)))**2)
    return arg_min

### The multi-relation matrix factorization (MRMF) corrdinate decent
    algorithm ######
def
    MRMF_CD(X,Y,Z,alpha_x,alpha_y,alpha_z,lambda_u,lambda_v,lambda_c,lambda_b,k,y):

    m = len(X) # Around 17400
    n = len(X[0]) ### Context features length
    c = len(Y[0]) ### Features length
    l =len(Z[0]) ### broad words length
    U,V,C,B = random_matrices(m,n,c,l,k)
    I = np.identity(k)
    arg_min = 1
    arg_min_old = 2

    while np.fabs(arg_min_old - arg_min) > 0.2:
        U = np.dot((np.dot(np.array(float(alpha_x))*X,V) +
            np.dot(np.array(float(alpha_y))*Y,C) +
            np.dot(np.array(float(alpha_z))*Z,B)) ,
            LA.pinv(np.dot(np.array(float(alpha_x))*V.T,V) +
            np.dot(np.array(float(alpha_y))*C.T,C)+
            np.dot(np.array(float(alpha_y))*B.T,B) +
            np.array(float(lambda_u))*I))
        V = np.dot(LA.pinv(np.dot(np.array(float(alpha_x))*U.T,U) +
            np.array(float(lambda_v))*I),
            np.dot(np.array(float(alpha_x))*U.T,X)).T
        B = np.dot(LA.pinv(np.dot(np.array(float(alpha_z))*U.T,U) +
            np.array(float(lambda_v))*I),
            np.dot(np.array(float(alpha_z))*U.T,Z)).T
        C = np.dot(LA.pinv(np.dot(np.array(float(alpha_y))*U.T,U) +
            np.array(float(lambda_c))*I),
            np.dot(np.array(float(alpha_y))*U.T,Y)).T
        arg_min_old = arg_min
        arg_min = L2_loss(X,Y,Z,U,V,C,B,alpha_x, alpha_y, alpha_z, lambda_u,
            lambda_v, lambda_c, lambda_b, y)
    return U,V,C,B

def predict(X_test,Z_test,U,V,B,C,alpha_x,alpha_z,lambda_u,I):
    U_test = alpha_x*np.dot(np.dot(X_test,V),LA.pinv(np.dot(V.T,V))) +
        alpha_z*np.dot(np.dot(Z_test,B),LA.pinv(np.dot(B.T,B)))
    Y_test = np.dot(U_test,C.T)
    return Y_test,U_test

def performance_test(X_test_data,Y_test,Y_test_result):
    performace = 0
    num_tests = 0
    for i in range(len(list(Y_test))):
        num_tests +=1
        if X_test_data.keys()[i] in Y_test_result.keys():
```

```
            if list(Y_test[i]).index(max(list(Y_test[i]))) ==
                Y_test_result[X_test_data.keys()[i]].index(
                max(Y_test_result[X_test_data.keys()[i]])):
              performance = performance + 1

    return num_tests, performance, (performance/float(len(list(Y_test))))*100

def
    dir_reading(x_tr_path,x_ts_path,y_tr_path,y_ts_path,z_tr_path,z_ts_path,alpha_x,alpha_z,k):

    ###### Reading the training and test set files from a directory and sort
        #####
    X_train_path = sorted(os.listdir(x_tr_path))
    X_test_path = sorted(os.listdir(x_ts_path))
    Y_train_path = sorted(os.listdir(y_tr_path))
    Y_test_path = sorted(os.listdir(y_ts_path))
    Z_train_path = sorted(os.listdir(z_tr_path))
    Z_test_path = sorted(os.listdir(z_ts_path))
    num_words = 0 #### counting the number of test words to evaluate the
        model
    corr = 0     #### counting the number of correctly classified test words

    ##### 10-fold cross validation ########
    for cv in range(0,10):

        #### Integrating the directory path and file name to read the file
            #####
        X = str(x_tr_path + X_train_path[cv])
        Y =  str(y_tr_path + Y_train_path[cv])
        Z =  str(z_tr_path + Z_train_path[cv])
        X_test = str(x_ts_path + X_test_path[cv])
        Y_test_result = str(y_ts_path + Y_test_path[cv])
        Z_test = str(z_ts_path + Z_test_path[cv])

        ######## Alpha Y and Regularizer parameteres value ########
        alpha_y = 1
        lambda_u = 0.0001
        lambda_v = 0.0001
        lambda_c = 0.0001
        lambda_b = 0.0001

        ######### Reading files ###################
        X_data = data_reading(X)
        Y_data = data_reading(Y)
        y =  construct_y(Y_data) ###### To make Y convenient for computaton
            by a dictionary
        Z_data = data_reading(Z)
        X_test_data = data_reading(X_test)
        Z_test_data = data_reading(Z_test)
        Y_test_result_data = data_reading(Y_test_result)

        ###### Calling MRMF #########
        U,V,C,B = MRMF_CD(X_data.values(),Y_data.values(),Z_data.values(),
            float(alpha_x), float(alpha_y), float(alpha_z), float(lambda_u),
            float(lambda_v), float(lambda_c), float(lambda_b),k,y)
        I = np.identity(k) ###### Identity matrix
```

```
        Y_test,U_test = predict(X_test_data.values(), Z_test_data.values(),
            U, V, B, C, alpha_x, alpha_z, lambda_u, I) ### Predict Y test
            set classification
        num_tests, performance,perc =
            performance_test(X_test_data,Y_test,Y_test_result_data) ### Test
        corr = corr + performance
        num_words = num_words + num_tests

    co = float(corr/float(num_words )) * 100
    print ('Finally, correctly categorized words : ',corr, ' Lambda
        ',lambda_u,' alpha x ',float(alpha_x),' alpha z ',float(k),' In
        percentage : ', float(corr/float(num_words )) * 100)

def __init__():
 for k in [50, 100, 200,300, 400]:
    for alpha_x in [0.001,0.0001,0.00001]:
        for alpha_z in [0.0003, 0.0001,0.00001]:
            dir_reading('\X_train\\', '\X_test\\', '\Y_train\\', '\Y_test\\',
                '\Z_Train\\', '\Z_Test\\', alpha_x, alpha_z, k)
```

A.2 MRMF-SGD for Semantic Similarity

This appendix shows the MRMF-SGD script which has used to train a model for semantic similarity task. The script works for different datasets, but the following script is an example on CogALex2016 shared task dataset. It uses SGD algorithm to optimize the matrices. This script can be modified for the phrasal semantics as well, because, the phrasal semantic task has used MRMF with SGD as well.

```
import numpy as np
import math
import random
import sys

def similarity_mf(w_1,w_2):
        try:
            n_1 = words.index(w_1)
            n_2 = words.index(w_2)
            return np.inner(np.array(U[n_1]),np.array(U[n_2]))
        except:
            return 0

## To train the threshold/split which says semantically related or not ##
def rank(data,simfun):
        ranked_list = []
        for w_1, w_2, v in data:
            ranked_list.append((w_1,w_2,v,simfun(w_1,w_2)))
        return sorted(ranked_list, key=lambda x: x[3],reverse=True)

def f1_of_rank(rank):
        th_f1 = []
        true_pos = 0
        false_pos = 0
        false_neg = len([a for (a,b,V,t) in rank if V])
        for w_1,w_2,trueval,th in rank:
            if trueval:
```

```
                true_pos += 1
                false_neg -= 1
            else:
                false_pos +=1
            if true_pos:
                prec = float(true_pos)/float(true_pos+false_pos)
                rec = float(true_pos)/float(true_pos+false_neg)
                f1 = 2*prec*rec/(prec+rec)
            else:
                f1 = 0

            if len(th_f1) > 0 and th == th_f1[-1][0]:
                if f1 > th_f1[-1][1]:
                    th_f1[-1] = (th,f1)
            else:
                th_f1.append((th,f1))

        return zip(*th_f1)

#################### Predict the similartiy of the words #######
def rank_pred(data,simfun,th):
        ranked_list = []
        ranked_pred = []
        for w_1, w_2, v in data:
            n_1 = words.index(w_1)
            n_2 = words.index(w_2)
            ranked_list.append((w_1,w_2,v,Y_test[n_1][n_2]))
            treshold_loc = sum(Y_test[n_1])/float(len(Y_test[n_1]))
            if float(Y_test[n_1][n_2]) >= float(treshold_loc):
                ranked_pred.append((w_1,w_2,v,True))
            else:
                ranked_pred.append((w_1,w_2,v,False))
        return ranked_pred

#################### Evaluate the predicted results #################
def evaluate(pairs):
        true_pos = 0
        false_pos = 0
        false_neg = 0
        for w_1,w_2,trueval,predval in pairs:
            if trueval and predval:
                true_pos += 1
            elif trueval and not predval:
                false_neg += 1
            elif not trueval and predval:
                false_pos +=1

        if true_pos:
            prec = float(true_pos)/float(true_pos+false_pos)
            rec = float(true_pos)/float(true_pos+false_neg)
            f1 = 2*prec*rec/(prec+rec)
        else:
            f1 = 0
        return f1

###### The SGD algorithm ###############
def SGD(X,Y,k, alpha_x, alpha_y,mu, lambda_u, lambda_v):
```

```
    m = len(X)
    n = len(X[0]) # Around 17400

    I = np.identity(k)
    stand_dev = np.std(I)
    U = np.random.normal(0, stand_dev, size=(m,k))
    V = np.random.normal(0, stand_dev, size=(n,k))

    ####### The index of each value of the matrix #####
    loop_size =len(X)*len(X[0])

    itere = 0
    while itere < 60: #### Since the computation time is really slow, the
        convergernce has limited by limited number loops ##
        itere += 1
        index_loop = 0
        while itere <= loop_size:

            i = random.choice(list(range(0,len(X))))
            j = random.choice(list(range(0,len(X[0]))))
            prediction_ij = np.dot(U[i].T , V[j])
            eij = X[i][j] - alpha_x*prediction_ij
            U[i] = np.array(U[i]) + mu * (2 * eij * np.array(V[j]) + lambda_u
                * np.array(U[i]))
            V[j] = np.array(V[j]) + mu * (2 * eij * np.array(U[i]) + lambda_v
                * np.array(V[j]))

            i = random.choice(list(range(0,len(X))))
            if (i,j) in train_dict.keys():
                    eij_y = Y[i][j] - alpha_y*(np.dot(U[i].T , U[j]))
                    U[i] = np.array(U[i]) + mu * (2 * eij_y * np.array(V[j])
                        + lambda_u * np.array(U[i]))
            index_loop += 1

        ### L2 loss objective function. Error rate computation ###
        error = 0.0
        for i in range(m):
            for j in range(n):
                prediction_ij = np.dot(U[i].T , V[j])
                error += (X[i][j]-prediction_ij)**2
        error = math.sqrt(error)
        print(error)
        mu = 0.95*mu
    return U, V

########## Read the experements dataset ############
def read_data(fname):
        instances = []
        f_train = open(fname)
        data_dict={}
        for line in f_train:
            w_1,w_2,sim = line.strip().split('\t')
            n_1 = words.index(w_1)
            n_2 = words.index(w_2)
            data_dict[(n_1,n_2)] = sim
            if(sim == 'TRUE'):
```

```
                instances.append([w_1,w_2,True])
            else:
                instances.append([w_1,w_2,False])
        f_train.close()
        return instances,data_dict

####### Read file and return matrix and the list of words ##########
def read_ppmi_matrix(file_name):
    matrix = []
    wordlist = []
    f = open(file_name)
    for line in f:
        word, v_as_string = line.strip().split('\t')
        matrix.append(list(map(float,v_as_string.split(','))))
        wordlist.append(word)
    f.close()
    return wordlist, matrix

######## Reading the ppmi matrix from a file ##########
words, X = read_ppmi_matrix('All_train_test_X.txt')
words, Y = read_ppmi_matrix('All_train_test_Y.txt')

######## Reading the training and test sets from a file ##########
traindata,train_dict = read_data(r'gold_task1_train.txt')
testdata,test_dict = read_data(r'gold_task1_test.txt')
for k in [50, 100, 200, 300, 400]:
    for mu in [0.1, 0.01, 0.001, 0.0001, 0.00001]:
        U,V = SGD(X,Y,int(k),float(1.0),float(1.0), float(mu),0.0015,0.0015)
Y_test = np.dot(U,U.T)

###### Call rank and f1_of_rank functions to train the threshold/split ####
rl = rank(traindata,similarity_mf)
ths_tr,fscores_tr = f1_of_rank(rl)

## Extract the best threshold which has the higest F1Score in the training
    set ##
thre_fscore_tr =
    float(ths_tr[fscores_tr.index(float(max(list(fscores_tr))))])

###### test set ########
pred_data_fi = rank_pred(testdata,similarity_mf,thre_fscore_tr)
real_f1_ts_fi = evaluate(pred_data_fi)
```

Bibliography

Abulaish, Muhammad and Tarique Anwar (2013). "A keyphrase-based tag cloud generation framework to conceptualize textual data". In: *International Journal of Adaptive, Resilient and Autonomic Systems (IJARAS)* 4.2, pp. 72–93.

Aga, Rosa Tsegaye and Christian Wartena (2015). "Constructing Concept Clouds from Company Websites". In: *Proceedings of the 15th International Conference on Knowledge Technologies and Data-driven Business*. i-KNOW '15. Graz, Austria: ACM, 38:1–38:4. ISBN: 978-1-4503-3721-2. DOI: 10.1145/2809563.2809615. URL: http://doi.acm.org/10.1145/2809563.2809615.

Aga, Rosa Tsegaye, Christian Wartena, and Michael Franke-Maier (2016). "Automatic Recognition and Disambiguation of Library of Congress Subject Headings". In: *TPDL*. Vol. 9819. Lecture Notes in Computer Science. Springer, pp. 442–446.

Aga, Rosa Tsegaye et al. (2016a). "Integrating Distributional and Lexical Information for Semantic Classification of Words using MRMF". In: *COLING*. ACL, pp. 2708–2717.

Aga, Rosa Tsegaye et al. (2016b). "Learning Thesaurus Relations from Distributional Features". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Ed. by Nicoletta Calzolari (Conference Chair) et al. Portorož, Slovenia: European Language Resources Association (ELRA). ISBN: 978-2-9517408-9-1.

Aga, Rosa Tsegaye et al. (2017). "Automatic Identification of Synonym Relations in the Dutch Parliament's Thesaurus". en. In: *Archives of Data Science, Series A*. Archives of Data Science, Series A 2017.2(1). ISSN: 2363-9881. DOI: 10.5445/KSP/1000058749/23. URL: http://nbn-resolving.de/urn:nbn:de:swb:90-734255.

Attia, Mohammed et al. (2016). "CogALex-V Shared Task: GHHH - Detecting Semantic Relations via Word Embeddings". In: *CogALex@COLING*. The COLING 2016 Organizing Committee, pp. 86–91.

Bastian, Mathieu, Sebastien Heymann, and Mathieu Jacomy (2009). "Gephi: An Open Source Software for Exploring and Manipulating Networks". In: URL: http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154.

Battig, W.F. and W.E. Montague (1969). *Category norms for verbal items in 56 categories: a replication and extension of the Connecticut category norms*. Journal of experimental psychology monograph. American Psychological Association. URL: https://books.google.de/books?id=bYwhAQAAMAAJ.

Biemann, Chris (2005). "Ontology Learning from Text: A Survey of Methods." In: *LDV forum*. Vol. 20. 2, pp. 75–93.

Bullinaria, John A. and Joseph P. Levy (2007). "Extracting semantic representations from word co-occurrence statistics: A computational study". In: *Behaviour Research Methods* 39.3, pp. 510–526.

— (2012). "Extracting Semantic Representations from Word Co-occurrence Statistics: Stop-lists, Stemming and SVD". In: *Behaviour Research Methods* 44.3, pp. 890–907.

Camacho-Collados, José, Mohammad Taher Pilehvar, and Roberto Navigli (2015). "NASARI: a Novel Approach to a Semantically-Aware Representation of Items". In: *NAACL*.

Campos, Luis M. De and Alfonso E. Romero (2010). "Bayesian network models for hierarchical text classification from a thesaurus". In: *Int. J. Approx. Reasoning*, pp. 932–944.

Chang, Chih-Chung and Chih-Jen Lin (May 2011). "LIBSVM: A Library for Support Vector Machines". In: *ACM Trans. Intell. Syst. Technol.* 2.3, 27:1–27:27. ISSN: 2157-6904. DOI: 10.1145/1961189.1961199. URL: http://doi.acm.org/10.1145/1961189.1961199.

Chersoni, Emmanuele, Giulia Rambelli, and Enrico Santus (2016). "CogALex-V Shared Task: ROOT18". In: *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon (CogALex - V)*. Osaka, Japan: The COLING 2016 Organizing Committee, pp. 98–103. URL: http://www.aclweb.org/anthology/W16-5313.

Crouch, Carolyn J. (1990). "An approach to the automatic construction of global thesauri". In: *Information Processing & Management* 5, pp. 629–640.

Crouch, Carolyn J. and Bokyung Yang (1992). "Experiments in automatic statistical thesaurus construction". In: *SIGIR '92: Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 77–88.

Cruys, Tim Van de, Thierry Poibeau, and Anna Korhonen (2013). "A tensor-based factorization model of semantic compositionality". In: *Proceedings of the Conference of the North American Chapter of the Association of Computational Linguistics (HTL-NAACL)*, pp. 1142–1151.

Cui, Weiwei et al. (2010). "Context preserving dynamic word cloud visualization". In: *Pacific Visualization Symposium (PacificVis), 2010 IEEE*. IEEE, pp. 121–128.

Cunningham, Hamish et al. (2002). "Gate: an Architecture for Development of Robust HLT Applications". In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 168–175.

Curran, James R., Stephen Clark, and Johan Bos (2007). "Linguistically Motivated Large-scale NLP with C&C and Boxer". In: *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. ACL '07. Prague, Czech Republic: Association for Computational Linguistics, pp. 33–36. URL: http://dl.acm.org/citation.cfm?id=1557769.1557781.

Curran, James R. and Marc Moens (2002). "Improvements in Automatic Thesaurus Extraction". In: *Unsupervised Lexical Acquisition: Proceedings of the Workshop of the ACL Special Interest Group on the Lexicon (SIGLAX)*, pp. 59–66.

Drumond, Lucas and Rosario Girardi (2008). "A Survey of Ontology Learning Procedures." In: *WONTO* 427.

Drumond, Lucas Rego et al. (2014). "Optimizing Multi-Relational Factorization Models for Multiple Target Relations". In: *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. CIKM '14. Shanghai, China: ACM, pp. 191–200. ISBN: 978-1-4503-2598-1. DOI: 10.1145/2661829.2662052. URL: http://doi.acm.org/10.1145/2661829.2662052.

Eijck, Jan van and Christina Unger (2010). *Computational Semantics with Functional Programming*. 1st. New York, NY, USA: Cambridge University Press. ISBN: 0521760305, 9780521760300.

Evert, Stefan (2016). "CogALex-V Shared Task: Mach5 – A traditional DSM approach to semantic relatedness". In: *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon (CogALex - V)*. Osaka, Japan: The COLING 2016 Organizing Committee, pp. 92–97. URL: http://www.aclweb.org/anthology/W16-5312.

Fan, Rong-En et al. (June 2008). "LIBLINEAR: A Library for Large Linear Classification". In: *J. Mach. Learn. Res.* 9, pp. 1871–1874. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=1390681.1442794.

Fillmore, Charles J. (1982). "Frame semantics". In: *Linguistics in the Morning Calm*. Seoul, South Korea: Hanshin Publishing Co., pp. 111–137.

Finkelstein, Lev et al. (2001). "Placing search in context: the concept revisited". In: *Proceedings of the Tenth International World Wide Web Conference, WWW 10, Hong Kong, China, May 1-5, 2001*, pp. 406–414. DOI: 10.1145/371920.372094. URL: http://doi.acm.org/10.1145/371920.372094.

Firm, Nick Riemer 1972-ProQuest (2010). *Introducing semantics*. Cambridge ; New York : Cambridge University Press.

Firth, J. R. (1957). "A synopsis of linguistic theory 1930-55." In: 1952-59, pp. 1–32.

Frank, Eibe et al. (1999). "Domain Specific Keyword Extraction". In: *IJCAI '99 Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, p. 686.

Gambette, Philippe and Jean Véronis (2010). "Visualising a text with a tree cloud". In: *Classification as a Tool for Research*. Springer, pp. 561–569. URL: \url{http://hal-lirmm.ccsd.cnrs.fr/lirmm-00373643/document}.

Garcia-Alsina, Montserrat, Christian Wartena, and Sönke Lieberam-Schmidt (2015). "Challenges to Construct Regional Knowledge Maps for Territories' Sustainable Development". In: *Knowledge Discovery, Knowledge Engineering and Knowledge Management*. Ed. by Ana Fred et al. Vol. 454. Communications in Computer and Information Science. Springer Berlin Heidelberg, pp. 387–399. ISBN: 978-3-662-46548-6. DOI: \url{10.1007/978-3-662-46549-3\textunderscore25}. URL: \url{http://dx.doi.org/10.1007/978-3-662-46549-3\textunderscore25}.

Gazendam, Luit, Christian Wartena, and Rogier Brussee (2010a). "Thesaurus Based Term Ranking for Keyword Extraction". In: *Proceedings*. Ed. by A. Min Tjoa and Roland R. Wagner. Los Alamitos, Calif: IEEE Computer Society, pp. 49–53. ISBN: ISBN 978-0-7695-4174-7.

— (2010b). "Thesaurus Based Term Ranking for Keyword Extraction". In: *Database and Expert Systems Applications, DEXA, International Workshops, Bilbao, Spain, August 30 - September 3, 2010*. Ed. by A. Min Tjoa and Roland R. Wagner. IEEE Computer Society, pp. 49–53. ISBN: ISBN 978-0-7695-4174-7.

Giesbrecht, Eugenie (2010). "Towards a Matrix-based Distributional Model of meaning". In: *Proceedings of the NAACL HLT 2010 Student Research Workshop*. Los Angeles, California: ACL, pp. 23–28.

Gross, Tina and Arlene G. Taylor (2005). "What Have We Got to Lose? The Effect of Controlled Vocabulary on Keyword Searching Results". In: *College & Research Libraries* 66.3, pp. 212–230. DOI: 10.5860/crl.66.3.212.

Gross, Tina, Arlene G. Taylor, and Daniel N. Joudry (2015). "Still a Lot to Lose: The Role of Controlled Vocabulary in Keyword Searching". In: *Cataloging & Classification Quarterly* 53.1, pp. 1–39. DOI: 10.1080/01639374.2014.917447.

Hagiwara, Masato (2008). "A Supervised Learning Approach to Automatic Synonym Identification Based on Distributional Features for Computational Linguistics, June 15-20, 2008, Columbus, Ohio, USA, Student Research Workshop". In: *ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pp. 1–6. URL: \url{http://www.aclweb.org/anthology/P08-3001}.

Harris, Zellig S. (1954). "Distributional structure". In: *Word* 10.23, pp. 146–162.

Hassan-Montero, Yusef and Victor Herrero-Solana (2006). "Improving tag-clouds as visual information retrieval interfaces". In: *International Conference on Multidisciplinary Information Sciences and Technologies*, pp. 25–28.

Hirsch, Laurie and David Tian (2013). "Txt2vz: a new tool for generating graph clouds". In: *Conceptual Structures for STEM Research and Education*. Springer, pp. 322–331.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780.

Hovy, Eduard and Chin-Yew Lin (1998). "Automated Text Summarization and the SUMMARIST System". In: *Proceedings of a Workshop on Held at Baltimore, Maryland: October 13-15, 1998*. TIPSTER '98. Baltimore, Maryland: Association for Computational Linguistics, pp. 197–214. DOI: 10.3115/1119089.1119121. URL: https://doi.org/10.3115/1119089.1119121.

Hsu, Chih-Wei, Chih-Chung Chang, and Chih-Jen Lin (2003). "A Practical Guide to Support Vector Classification". In:

Jacomy, Mathieu et al. (2011). "Forceatlas2, a continuous graph layout algorithm for handy network visualization". In: *Medialab center of research* 560.

Jurafsky, Daniel and James H. Martin (2000). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. 1st. Upper Saddle River, NJ, USA: Prentice Hall PTR. ISBN: 0130950696.

Karlgren, Jussi and Magnus Sahlgren (2001). "Vector-based Semantic Analysis using Random Indexing and Morphological Analysis for Cross-Lingual Information Retrieval". In: *CLEF (Working Notes)*. Vol. 1167. CEUR Workshop Proceedings. CEUR-WS.org.

Kaser, Owen and Daniel Lemire (2007). "Tag-cloud drawing: Algorithms for cloud visualization". In: *arXiv preprint cs/0703109*.

Keith, Jeff, Chris Westbury, and James Goldman (2015). "Performance impact of stop lists and morphological decomposition on word–word corpus-based semantic space models". In: *Behavior Research Methods* 47.3, pp. 666–684. ISSN: 1554-3528. DOI: 10.3758/s13428-015-0614-z. URL: http://dx.doi.org/10.3758/s13428-015-0614-z.

Kiela, Douwe and Stephen Clark (2014). "A Systematic Study of Semantic Vector Space Model Parameters". In: *2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*. Stroudsburg, PA, USA: Association for Computational Linguistics, pp. 21–30.

Kim, Su Nam et al. (2013). "Automatic keyphrase extraction from scientific articles". In: *Language Resources and Evaluation* 47.3, pp. 723–742. DOI: 10.1007/s10579-012-9210-3.

Korkontzelos, I. et al. (2013). "SemEval-2013 Task 5: Evaluating Phrasal Semantics." In: *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2013)*. URL: http://www.ukp.tu-darmstadt.de/fileadmin/user_upload/Group_UKP/publikationen/2013/SemEval2013-Task5.pdf.

Lapesa, Gabriella, Stefan Evert, and Sabine Schulte im Walde (2014). "Contrasting Syntagmatic and Paradigmatic Relations: Insights from Distributional Semantic Models". In: *Proceedings of the 3rd Joint Conference on Lexical and Computational Semantics*. Dublin, Ireland, pp. 160–170.

Lee, Lillian (2000). "Measures of Distributional Similarity". In: *CoRR* cs.CL/0001012. URL: http://arxiv.org/abs/cs.CL/0001012.

Lenci, Alessandro and Giulia Benotto (2012). "Identifying Hypernyms in Distributional Semantic Spaces". In: *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*. SemEval '12. Montreal, Canada: Association for Computational

Linguistics, pp. 75–79. URL: http://dl.acm.org/citation.cfm?id=2387636.2387650.

Lippert, Christoph et al. (2008). "Relation-Prediction in Multi-Relational Domains using Matrix-Factorization". In: *NIPS 2008 Workshop: Structured Input - Structured Output*.

Luce, Kanan, Jiaxing Yu, and Shu-Kai HSIEH (Dec. 2016). "CogALex-V Shared Task: LOPE". In: *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon (CogALex - V)*. Osaka, Japan: The COLING 2016 Organizing Committee, pp. 110–113. URL: https://www.aclweb.org/anthology/W16-5315.

Medelyan, Olena and Ian H. Witten (2005). "Thesaurus-based index term extraction for agricultural documents". In: *Proc. of the 6th Agricultural Ontology Service workshop*.

Meusel, Robert et al. (2010). "Thesaurus Extension Using Web Search Engines". In: *The Role of Digital Libraries in a Time of Global Change*, pp. 198–207. DOI: 10.1007/978-3-642-13654-2_24.

Mikolov, Tomas et al. (2013a). "Distributed Representations of Words and Phrases and their Compositionality". In: *NIPS*. Curran Associates, Inc., pp. 3111–3119. URL: http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

Mikolov, Tomas et al. (2013b). "Efficient Estimation of Word Representations in Vector Space". In: *CoRR* abs/1301.3781.

Mitchell, Jeff and Mirella Lapata (2008). "Vector-based Models of Semantic Composition". In: *Proceedings of ACL-08: HLT*. Columbus, Ohio, pp. 236–244.

Montague, Richard (Dec. 1970). "Pragmatics and intensional logic". In: 22.1–2, pp. 68–94. ISSN: 0039-7857 (print), 1573-0964 (electronic). DOI: https://doi.org/10.1007/BF00413599. URL: http://link.springer.com/article/10.1007/BF00413599.

Muskens, Reinhard (1996). *Combining Montague Semantics and Discourse Representation*. URL: http://cogprints.org/4715/.

Office for Official Publications of the European Communities (1995). *Thesaurus Eurovoc - Volume 2: Subject-Oriented Version*. Luxembourg. URL: http://europa.eu.int/celex/eurovoc.

Ogden, C.K. and I.A. Richards (1989). *The meaning of meaning: a study of the influence of language upon thought and of the science of symbolism*. International library of psychology, philosophy and scientific method. Harcourt Brace Jovanovich. URL: https://books.google.de/books?id=M6UQAQAAIAAJ.

Paynter, Gordon W (2005). "Developing practical automatic metadata assignment and evaluation tools for internet resources". In: *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*. ACM, pp. 291–300.

Pekar, Viktor, Michael Krkoska, and Steffen Staab (2004). "Feature weighting for co-occurrence-based classification of words". In: *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics, pp. 799 –806. DOI: \url{10.3115/1220355.1220470}.

Pennacchiotti, Marco et al. (2008). "Automatic Induction of FrameNet Lexical Units". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. EMNLP '08. Honolulu, Hawaii: Association for Computational Linguistics, pp. 457–465. URL: http://dl.acm.org/citation.cfm?id=1613715.1613773.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning (2014). "GloVe: Global Vectors for Word Representation". In: *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543. URL: http://www.aclweb.org/anthology/D14-1162.

Pouliquen, Bruno, Ralph Steinberger, and Camelia Ignat (2003). "Automatic annotation of multilingual text collections with a conceptual thesaurus". In: *Workshop in Ontologies and Information Extraction (EUROLAN2003)*.

Rubenstein, Herbert and John B. Goodenough (1965a). "Contextual correlates of synonymy". In: *Commun. ACM* 8.10, pp. 627–633.

— (1965b). "Contextual Correlates of Synonymy". In: *Commun. ACM* 8.10, pp. 627–633. ISSN: 0001-0782. DOI: \url{10.1145/365628.365657}. URL: \url{http://doi.acm.org/10.1145/365628.365657}.

Sahlgren, Magnus (2005). "An introduction to random indexing". In: *In Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE 2005*.

Salton, Gerard and Christopher Buckley (1988). "Term-weighting approaches in automatic text retrieval". In: *Information processing & management* 24.5, pp. 513–523.

Santus, Enrico et al. (2014). "Taking Antonymy Mask off in Vector Space". In: *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computing*. URL: http://www.aclweb.org/anthology/Y14-1018.

Santus, Enrico et al. (2016). "Nine Features in a Random Forest to Learn Taxonomical Semantic Relations". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Ed. by Nicoletta Calzolari (Conference Chair) et al. Portorož, Slovenia: European Language Resources Association (ELRA). ISBN: 978-2-9517408-9-1.

Schäfer, Roland and Felix Bildhauer (2012). "Building Large Corpora from the Web Using a New Efficient Tool Chain." In: *LREC*. Ed. by Nicoletta Calzolari et al. European Language Resources Association (ELRA), pp. 486–493. ISBN: 978-2-9517408-7-7. URL: http://dblp.uni-trier.de/db/conf/lrec/lrec2012.html#SchaferB12.

Shimizu, Nobuyuki et al. (2008). "Metric learning for synonym acquisition". In: *COLING '08 Proceedings of the 22nd International Conference on Computational Linguistics*, pp. 793–800.

Shwartz, Vered and Ido Dagan (2016). "CogALex-V Shared Task: LexNET - Integrated Path-based and Distributional Method for the Identification of Semantic Relations". In: *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon (CogALex - V)*. Osaka, Japan: The COLING 2016 Organizing Committee, pp. 80–85. URL: http://www.aclweb.org/anthology/W16-5310.

Spärck Jones, Karen (2004). "A statistical interpretation of term specificity and its application in retrieval". In: *Journal of documentation* 60, pp. 493–502.

Tsegaye, Rosa and Christian Wartena (2016). "CogALex-V Shared Task: HsH-Supervised – Supervised similarity learning using entry wise product of context vectors". In:

Tsuruoka, Yoshimasa et al. (2005). "Developing a Robust Part-of-speech Tagger for Biomedical Text". In: *Proceedings of the 10th Panhellenic Conference on Advances in Informatics*. PCI'05. Volos, Greece: Springer-Verlag, pp. 382–392. ISBN: 3-540-29673-5, 978-3-540-29673-7. DOI: 10.1007/11573036_36. URL: https://doi.org/10.1007/11573036_36.

Turney, Peter D. (2000). "Learning Algorithms for Keyphrase Extraction". In: *Inf. Retr.* 2.4, pp. 303–336.

— (2013a). "Distributional semantics beyond words: Supervised learning of analogy and paraphrase". In: *CoRR* abs/1310.5042. URL: http://arxiv.org/abs/1310.5042.

— (2013b). "Domain and Function: A Dual-Space Model of Semantic Relations and Compositions". In: *CoRR* abs/1309.4035. URL: http://arxiv.org/abs/1309.4035.

Turney, Peter D. and Patrick Pantel (2010). "From Frequency to Meaning: Vector Space Models of Semantics". In: *Journal of Artificial Intelligence Research* 37, pp. 141–188.

Viegas, Fernanda B, Martin Wattenberg, and Jonathan Feinberg (2009). "Participatory visualization with Wordle". In: *Visualization and Computer Graphics, IEEE Transactions on* 15.6, pp. 1137–1144.

Wang, Jinghua, Jianyi Liu, and Cong Wang (2007). "Keyword Extraction Based on PageRank". In: *Advances in Knowledge Discovery and Data Mining* 4426, pp. 857–864.

Wartena, Christian (2013). "HsH: Estimating semantic similarity of words and short phrases with frequency normalized distance measures". In: *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012), Atlanta, Georgia, USA*.

— (2014). "On the effect of word frequency on distributional similarity Germany, October 8-10, 2014". In: *Proceedings of the 12th Edition of the Konvens Conference, Hildesheim, Germany, October 8-10, 2014*. Ed. by Josef Ruppenhofer and Gertrud Faaß. Universitätsbibliothek Hildesheim, pp. 1–10. ISBN: 978-3-934105-46-1.

Wartena, Christian, Rogier Brussee, and Wout Slakhorst (2010). "Keyword extraction using word co-occurrence". In: *Database and Expert Systems Applications, DEXA, International Workshops, Bilbao, Spain, August 30 - September 3, 2010*. Ed. by A. Min Tjoa and Roland R. Wagner. IEEE Computer Society, pp. 54–58. ISBN: ISBN 978-0-7695-4174-7.

Wartena, Christian and Montserrat Garcia-Alsina (2015). "Keyword Extraction from Company Websites for the Development of Regional Knowledge Maps". In: *Knowledge Discovery, Knowledge Engineering and Knowledge Management*. Ed. by Ana Fred et al. Vol. 454. Communications in Computer and Information Science. Springer Berlin Heidelberg, pp. 96–111. ISBN: 978-3-662-46548-6. DOI: \url{10.1007/978-3-662-46549-3\textunderscore7}. URL: \url{http://dx.doi.org/10.1007/978-3-662-46549-3\textunderscore7}.

Wartena, Christian et al. (2007). "Apolda: A practical tool for semantic annotation". In: *Database and Expert Systems Applications, 2007. DEXA'07. 18th International Workshop on*, pp. 288–292.

Weeds, Julie, David J. Weir, and Diana McCarthy (2004). "Characterising Measures of Lexical Distributional Similarity". In: *COLING 2004, Proceedings of the 20th International Conference on Computational Linguistics*. DOI: 10.3115/1220355.1220501.

Weeds, Julie et al. (2014). "Learning to Distinguish Hypernyms and Co-Hyponyms". In: *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. Dublin, Ireland: Dublin City University and Association for Computational Linguistics, pp. 2249–2259. URL: http://www.aclweb.org/anthology/C14-1212.

Witschel, Hans Friedrich (2005). "Using decision trees and text mining techniques for extending taxonomies". In: *Proceedings of the Workshop on Learning and Extending Lexical Ontologies by Using Machine Learning Methods*, pp. 61–68.

Wright, Stephen J. (June 2015). "Coordinate Descent Algorithms". In: *Math. Program.* 151.1, pp. 3–34. ISSN: 0025-5610. DOI: 10.1007/s10107-015-0892-3. URL: http://dx.doi.org/10.1007/s10107-015-0892-3.

Yi, Kwan and Lois Mai Chan (2010). "Revisiting the syntactical and structural analysis of Library of Congress Subject Headings for the digital environment". In: *Journal of the american society for information science and technology* 61.4, pp. 677–687.

Yih, Wen-tau and Vahed Qazvinian (2012). "Measuring Word Relatedness Using Heterogeneous Vector Space Models". In: *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. NAACL HLT '12. Montreal, Canada: Association for Computational Linguistics, pp. 616–620. ISBN: 978-1-937284-20-6. URL: http://dl.acm.org/citation.cfm?id=2382029.2382130.

ZBW - Leibniz Information Centre for Economics (2014). *STW Thesaurus for Economics*. Kiel, Germany. URL: http://zbw.eu/stw/versions/latest/about.en.html.